

# **Maestro Command Reference Manual**

---

Version 7.5, April 2006

For inquiries about Maestro™ contact:

Schrödinger  
1500 SW First Avenue, Suite 1180  
Portland, OR 97201-5881  
503-299-1150  
503-299-4532 fax  
Email: [help@schrodinger.com](mailto:help@schrodinger.com)

---

Copyright © 2006 Schrödinger, LLC. All rights reserved.

Maestro, CombiGlide, Epik, Glide, Impact, Jaguar, Liaison, Lig-Prep, Phase, Prime, QikProp, QikFit, QikSim, QSite, and Strike are trademarks of Schrödinger, LLC. MacroModel and Schrödinger are registered trademarks of Schrödinger, LLC.

To the maximum extent permitted by applicable law, this document is provided “as is” without warranty of any kind. This document may contain trademarks of other companies.

Please note that any third party programs (“Third Party Programs”) or third party Web sites (“Linked Sites”) referred to in this document may be subject to third party license agreements and fees. Schrödinger, LLC and its affiliates have no responsibility or liability, directly or indirectly, for the Third Party Programs or for the Linked Sites or for any damage or loss alleged to be caused by or in connection with use of or reliance thereon. Any warranties that we make regarding our own products and services do not apply to the Third Party Programs or Linked Sites, or to the interaction between, or interoperability of, our products and services and the Third Party Programs. Referrals and links to Third Party Programs and Linked Sites do not constitute an endorsement of such Third Party Programs or Linked Sites.

Maestro Version 75112  
April 2006

# 1 The Maestro Command Language

This manual contains a listing of all supported Maestro commands, their functions, and their usage options. Use Maestro commands by entering them in the Command Input Area of the Main Application Window.

The Maestro command language syntax is:

**keyword** [*options*] [*operand*]

For example:

**entryimport** *wsreplace=true all=true format=maestro* ligands.mae

In the example given, “entryimport” is the keyword. As this example illustrates, the first item entered must be a keyword, and this keyword must match a known keyword completely, i.e., to all characters. However, Maestro does have an alias facility, which allows users to map keywords to shorter names, for example, “entryimport” to simply “import”. See the “alias” command for details.

Command options are used to set values that generally correspond to states within the program. As the name “options” suggests, inclusion of these items in a Maestro command is not mandatory. However, any number of option *name=<value>* pairs can be specified. Options can be of type: Boolean (yes/no, y/n, t/f, and true/false are valid values), string, integer, or real.

Once an option is set, its value will persist for all subsequent commands until it is specified again. A command consisting only of a keyword and options will set the states corresponding to those option values, but will not perform any other action. For example, the command:

**entryimport** *format=maestro all=true*

will not import structures from any file. It will only set the format for subsequent file reading and specify that all structures contained in the subsequently specified file be imported. If the following command is then issued:

**entryimport** ligands.mae

all structures from the ligands.mae Maestro file will be imported.

The operand is the remaining element of a Maestro command. The type of operand required is dependent on the specific command. For “entryimport”, the operand must be a file name. For other commands, it may be one or more atom numbers, or a string describing arbitrary atom sets, specified using Maestro’s Atom Specification Language (ASL). For more information about ASL, see [Chapter 3 \[ASL\], page 5](#).



## 2 Conventions Used in this Manual

This manual uses a number of typographical conventions to describe the command language.

The names of keywords are printed in boldface:

**entryimport**

The names of options are printed in italic, and literal choices for that option are in normal roman:

*wsreplace=true*

Values which the user is to replace with appropriate values are represented in angle brackets:

**entryimport** <file\_name>

Where there is a choice of a number of values, these are shown separated by vertical bars:

**entryimport** *format=maestro|mmod|sd|pdb|mol2*

If a value is optional then it is enclosed in square brackets:

**print** [ <file\_name> ]

The following conventions are used to describe acceptable values for options:

<n> means an integer

<x> means a real number

yes|no means a boolean value (could also be true|false, y|n, on|off)

<text> represents a string value. If this contains spaces, it must be enclosed in double quotation marks

The following are some conventions used for operands:

<atom\_number>

represents an atom number

<atom1>

represents the first atom number in the operations, <atom2> the second, and so on

<ASL>

represents a valid string in the Atom Specification Language

<ESL>

represents a valid string in the Entry Specification Language



## 3 The Atom Specification Language

This chapter documents the syntax of Maestro’s Atom Specification Language (ASL).

### 3.1 Why an Atom Specification Language?

- To provide a flexible way to define sets of atoms in complex macromolecular systems
- To serve as the basis of a “Sets” facility in Maestro
- To allow atom specification from textual input which some users find faster than picking atoms from the main structure window.

### 3.2 The ASL Hierarchy

`entry > molecule > chain > residue > atom`

There are five classes which make up the atom specification language. Each is listed below.

In this section the minimum acceptable abbreviation is shown outside the square brackets. So for example `m[molecule]` means that the minimal acceptable abbreviation for `molecule` is `m`.

**e[ntry]** This is the top level class in the language. An entry is all atoms in the workspace associated with a single entry in the currently open project.

**m[olecule]**

The term molecule is used in the normal chemical sense meaning all atoms which are connected by a single covalent path.

**c[hain]**

This corresponds to a chain as specified in the PDB file format. Note that this chain may be a subset of a molecule, e.g., when chains are linked by disulphide linkages.

**r[esidue]**

An arbitrary collection of one or more covalently bound atoms within a molecule, such as the monomer units in a polymer.

**a[tom]** A single atom.

Each class is optional. If absent all entities of that type are matched.

### 3.3 Atom Specification

A complete specification is a class name and some property specified by a property name and property list. The syntax is:

*class.property* ⟨propertylist⟩

Items in a property list may be separated by comma, whitespace or both. Ranges (lower-upper) may be used where appropriate. Unterminated ranges are taken to include all available numbers. For example, if there are four molecules in the system then the specifications:

- mol. 2, 3, 4
- mol. >=2
- mol. 2-4
- mol. >1

are equivalent.

In a similar manner,

- mol. 1, 2, 3
- mol. <=3
- mol. 1-3
- mol. <4

are equivalent.

All names of properties and characters in property lists are treated in a case insensitive manner.

Wildcards are supported for atom and set names. A '\*' will match zero or more characters and a '?' will match any single character. You can include comments in a specification by placing a '#' character before the text you wish to hide.

In this section the minimum acceptable abbreviation is shown outside the square brackets. So, for example, a[tom].pt[ype] means that the minimal acceptable abbreviation for atom.ptype is a.pt.

e[ntry].

[name] Because entry names are the only entry properties specifiable using ASL expressions, the word `name` can be completely omitted, i.e., `entry.name` `ename` and `entry.ename` are equivalent. Note, however, that the '.' is still required even if the property name is not included. A valid property list for the 'entry name' property is a list of entry names. Wildcard characters are permitted. For example:

```
entry. e1  
entry.name recep, lig*
```

**m[molecule].**

[number] Because numbers are the only molecular properties specifiable using ASL expressions, the word **number** can be completely omitted, i.e., `mol.number 1` and `mol. 1` are equivalent. Note, however, that the ‘.’ is still required even if the property name is not included. A valid property list for the ‘molecule number’ property is a set of numbers or a range. For example:

```
mol. 1-4
mol. 1,2,3,4
```

**m[odulo]** a property which can be used to select every *n*th molecule. For example: `mol.mod 10 1` will select molecule 1, 11, 21, etc.

**e[ntrynum]**

a property which can be used to select molecules based on their entry-relative numbering. For example: `mol.entrynum 1` will select the first molecule in each entry.

**a[toms]**

a property which can be used to select molecules based on the number of atoms they contain. For example: `mol.atoms 200` will select molecules that contain exactly 300 atoms. Other examples are:

```
mol.atoms 200-300
mol.a > 200
```

**w[eight]**

a property which can be used to select molecules based on their molecular weight. For example: `mol.weight 200.12` will select the atoms that have a molecular weight of exactly 200.12. Other examples are:

```
mol.weight <=300.0
mol.weight > 200.0
```

**c[hain].**

This class designation allows you to specify atoms using chain attributes. Combine with the ‘name’ property.

**[name]**

Because names are the only chain properties that are specifiable by ASL, the word **name** can be omitted. For example, the specifications `chain.name A` and `chain. A` are equivalent. Note, however, that the ‘.’ is required even if the property name is not included. A valid property list for the ‘chain name’ property is a single character representing a PDB

chain name. Some examples of equivalent acceptable 'chain name' expressions are:

```
chain.name A  
chain. A  
c. A
```

**r[esidue]**

This ASL class designation allows you to specify atoms based on residue properties. Combine with one of the following property specifications.

**[name] or [number]**

Either residue names or numbers (but not both) can be used in property lists which do not specify a property name. For example, the following are valid ASL expressions:

```
res. ala val leu  
res. 1 2 3
```

and will return atoms which are either in alanine, valine, or leucine residues, or atoms in residues 1, 2 or 3, respectively. Residue numbers can be specified with a range, e.g., `residue. 1-4` and may include negative values or zero.

**pt[type]**

The three-letter PDB code for the residue. This is the default for non-numeric characters in the property list, so the expression `res. arg` and `res.ptype arg` are equivalent. A valid property list for 'ptype' is comprised of three-character tokens. For example:

```
res.ptype gly, val, ala  
res. gly val ala
```

**m[type]**

The one letter residue codes as used in Maestro. A valid property list for 'mtype' is comprised of one-character tokens. For example:

```
res.mtype g, v, a  
res.m g, v, a
```

**po[larity]**

The polarity of the residue. The property list must consist only of the following descriptor types:

**h[ydrophobic]**

returns atoms in hydrophobic residues

**pol[ar]** returns atoms in polar residues

**pos[itive]**

returns atoms in residues with positive formal charges

**n[egative]**

returns atoms in residues with negative formal charges

For example,

```
residue.polarity hydrophobic  
residue.pol pos,neg  
res.pol h pos neg
```

**sec[ondary\_structure]**

The secondary structure of the residue. The property list must consist only of the following descriptor types:

**h[helix]** returns atoms in helical regions

**s[trand]** returns atoms in strand regions

**l[oop]** returns atoms in loop regions

For example,

```
residue.sec helix  
residue.sec hel, str  
res.sec l, s
```

**pos[ition]**

The fractional position of the residue. The property list must include two real numbers representing a fractional range of residue numbers. For example, if there are 100 residues numbered from 1-100, the specification:

```
residue.pos 0.0 0.1  
will return residues 1 to 10.
```

**i[nscode]**

The insertion code of the residue. A property list should include one-character tokens representing insertion codes. For example:

```
residue.inscode a  
will get all residues with insertion code 'a', while:  
res. 25 and res.inscode b  
will get residue 25b.
```

**a[tom]** Atom ptypes and numbers may be mixed in property lists where no explicit property is specified. For example, the following is valid:

```
atom. 1,2,3,CA
```

and returns atoms 1,2 and 3 and any alpha carbons.

**pt[type]** The PDB atom names. A valid property list for 'ptype' consists of acceptable PDB names. This is the default property for non-numeric components of property lists and, as such, the word 'ptype' may be omitted. Note that the '.' is required. The following specifications are equivalent and return the "backbone" atoms in a structure:

```
atom.ptype N,CA,C,O  
atom. N,CA,C,O  
a. n,ca,c,o
```

Note: see below for a discussion of how PDB atom names are specified and matched.

Wildcards as described above can be applied to ptypes.

**na[me]** The atom names. The property list must contain valid atom names. A valid atom name could be a string of any length which:

1. must have at least one non-digit character
2. must not contain any control characters (ASCII value < 0x20)
3. must not contain spaces or equal signs, unless the name is quoted

Wildcards as described above can be applied to atom names. Examples:

```
atom.name the_36th_carbon  
atom.na C15, O:66, H-77  
atom.na C* (returns atoms with name  
starting with C)  
atom.nam ??0* (returns atoms whose  
name's 3rd character is '0')
```

**n[umber]** The atom numbering. The property list must be a list or a range of numbers. This is the default property for numeric components of property lists. The following expressions are equivalent and return the atoms numbered 1, 2, 3, and 4.

```
atom.num 1,2,3,4
a. 1 2 3 4
atom. 1-4
```

**mo[lnum]** The atom numbering in the “by molecule” scheme. The property list must be a list or a range of numbers. For example, the expression:

```
atom.molnum 1
```

returns the first atom in each molecule, while the specification:

```
atom.molnum 1-10
```

returns the first 10 atoms in each molecule.

**en[trynum]**

The atom numbering in the “by entry” scheme. The property list must be a list or a range of numbers. For example, the expression:

```
atom.entrynum 1
```

returns the first atom in each entry, while the specification:

```
atom.entrynum 1-10
```

returns the first 10 atoms in each entry.

**m[type]**

The Maestro atom type. A valid property list for the ‘mtype’ property consists of Maestro atom types. The following expression is valid and specifies sp2 carbons and oxygens.

```
atom.mtype C2,O2
```

**e[element]**

The element symbol for the atom. A valid property list for the element is a list of standard periodic table symbols. To define all carbons and oxygens:

```
atom.ele C,O
```

**att[achments]**

The number of bonds the atom has to it. The property list must be a number in the range 0-6, but greater than ‘>’, less than ‘<’, and equals ‘=’ signs may also be used. The expression:

```
atom.att 1
```

returns all terminal atoms. The specification:

```
atom.att <=2
```

returns all terminal atoms and other atoms with 2 or fewer bonds. The specification:

`atom.att 0`

returns all isolated atoms.

**ato[micnumber]**

The atomic number. The property list must contain numbers only. Ranges of integers and greater than ‘>’, less than ‘>’, and equals ‘=’ signs may also be used. The expression:

`atom.atomicnum 1`

returns all hydrogen atoms. The specification:

`atom.ato 1-6`

returns all atoms in the range H to C.

**c[charge]**

The partial charge on the atom. A valid property list contains a value or range of floating point values. The expression:

`atom.charge 0.400`

returns atoms with partial charges of 0.400. The expression:

`atom.charge -0.6--0.4`

returns atoms with partial charges -0.6 to -0.4,

`atom.charge <0.0`

returns atoms with negative partial charges, and

`atom.charge >=0.5`

returns atoms with charges of 0.5 or greater.

**f[ormalcharge]**

The formal charge on the atom. A valid property list contains a value or range of integer values. The expression:

`atom.formal 0`

returns atoms with formal charges of 0. The expression:

`atom.formal -2 -- 1`

returns atoms with formal charges -2 to -1,

`atom.formal <0`

returns atoms with negative formal charges, and

`atom.formal >=1`

returns atoms with formal charges of 1 or greater.

**d[isplayed]**

Whether or not the atom is currently displayed in Maestro. No property list is used. For example, the expressions:

```
atom.displayed returns the set of all
displayed atoms
not atom.disp returns all atoms not
currently displayed
sidechain and fillres atom.disp
returns the side chains of residues where
at least one atom is displayed.
```

Using generalized atom properties. Some structures may have additional properties available. These are referenced directly by their data names appended onto the "atom." class. These properties are either of integer, real, boolean or string type and the datanames are encoded as beginning with i\_, r\_, b\_ or s\_ respectively. It is possible to use these atom properties in conjunction with any other ASL expression. Any atoms which don't have these properties associated with them will never match. Some examples of using the ASL to address these properties are:

```
atom.i_my_integer_prop      1-4      atom.b_my_boolean_prop
atom.r_my_real_prop < 4.0 atom.s_my_string_prop LIG.
```

## 3.4 Operators

A number of operators are supported:

### 3.4.1 The Boolean and Operator

Boolean AND (set intersection). The syntax for this operation is:

```
spec1 and spec2
```

where **spec1** and **spec2** are valid atom specifications. This operation will return the set of atoms which meets the specifications **spec1** and **spec2**. For example, the expression:

```
mol. 1 and atom. CA
```

will return the set of all the alpha carbons of molecule 1. The specification:

```
res.num 1-100 and res. ala
```

returns all alanines in residues with numbers in the range 1-100.

### **3.4.2 The Boolean or Operator**

The Boolean OR operator. The syntax for this operations is:

```
spec1 or spec2
```

where **spec1** and **spec2** are valid atom specifications. This returns the set of atoms which meet either specification **spec1** or **spec2**. For example, the expression:

```
mol. 1 or atom.ptype CA
```

returns the set of all atoms that are in molecule number 1, or are alpha carbons. The specification:

```
res.num 1-100 or res.ptype ala
```

returns all residues either with numbers in the range 1-100, or any alanines.

### **3.4.3 The Boolean not Operator**

The Boolean NOT operator. The syntax for this operation is:

```
not spec1
```

where **spec1** is a valid atom specification. This returns the set of atoms that are not part of those defined by **spec1**. For example, the expression:

```
not atom. CA,C,N,O
```

will return a set containing all side chain atoms.

### **3.4.4 The fillres and fillmol Operators**

Two special operations, **fillres** and **fillmol**, can be used to “fill out” the atoms defined by a atom specification to complete residue or molecule boundaries. For example:

```
fillres atom.num 1,100,40
```

will return all the atoms in residues of which atoms 1,100 and 40 are members. In a similar way:

```
fillmol atom.num 1,100,40
```

will return all the atoms in molecules of which atoms 1,100 and 40 are members.

### **3.4.5 The within and beyond Operators**

The operators **within** and **beyond** can be used to define sets of atoms based on their distance from a set of atoms defined defined by an atom specification. The syntax for these operators is:

**within distance spec1**

which returns the atoms within, i.e, less than or equal to the distance in Angstroms of the set defined by spec1, and

**beyond distance spec2**

which returns the atoms which are further than the distance in Angstroms from the set defined by spec2.

For example, the expression:

**within 5.0 mol. 1**

returns the set of all atoms that are within 5 Å of molecule 1. The expression:

**beyond 5.0 mol. 2**

returns all atoms that are farther than 5 Å from molecule 2.

The combination of **fillres** and **within** or **beyond** is especially powerful.

**fillres within 5.0 mol. 1**

will produce a set containing the atoms of all complete residues that have atoms within 5 Å of molecule 1. Note that the **within** operator will also return the reference set of atoms:

**within 5.0 mol. 1**

returns the reference set of all atoms that are within 5 Å of molecule 1 and those that are part of molecule 1.

The **and** operator, when used with **within** and **beyond**, can be used to allow more specificity:

**mol. 2 and within 5.0 mol. 1**

returns the set of all atoms of molecule 2 that are within 5 Å of molecule 1.

### 3.4.6 The withinbonds Operator

A special type of **within**, one that finds all atoms within a certain number of bonds of the reference set:

**withinbonds <num\_bonds> <spec>**

For example:

**withinbonds 4 atom. 1**

will find all the atoms that are within four bonds of atom 1.

### 3.4.7 The beyondbonds Operator

A special type of **within**, one that finds all atoms beyond a certain number of bonds of the reference set:

**beyondbonds <num\_bonds> <spec>**

For example:

```
beyondbonds 4 atom. 1
```

will find all the atoms that are in the same molecule as atom 1 but beyond four bonds of atom 1.

## 3.5 Operator Priority

The order of priority of operators is (in decreasing order):

- `not`/`fillres`/`fillmol`
- `and`/`or`
- `within`/`beyond`

At equal levels of priority the expression will be evaluated left to right.

Examples:

```
within 5.0 mol. 1 or mol. 2
```

returns the set of all atoms that are within 5.0 Å of either molecule 1 or molecule 2 (`or` has higher priority).

```
not atom.ptype CA,C,O,N and mol. 1
```

returns the side chain atoms of molecule 1.

```
atom.ptype CA or mol. 1 and not res.pol polar
```

returns all alpha carbons and atoms in hydrophobic residues of molecule 1.

Parentheses can be used to override the order of evaluation:

```
not (atom.ptype CA,C,O,N or mol. 1)
```

produces all atoms either not in the backbone or not in molecule 1.

## 3.6 Implicit Operators

When no operator is specified, the following operations are assumed:

- Specifications within a property list have a Boolean `or` relationship. So the following specification:

```
atom.ptype CA,CB
```

matches any atom which has a PDB name of either CA or CB.

- When the classes are in order of decreasing priority in the class hierarchy, the `and` operator is assumed:

```
mol. 1 chain. A atom.ptype CA
```

is equivalent to:

```
mol. 1 and chain. A and atom.ptype CA  
and both return the alpha carbons in chain A of molecule 1.
```

## 3.7 Creating New Sets from Existing Ones

Maestro supports a mechanism for defining and naming atom sets via its “Sets” panel. The names of existing sets may be used in expressions if they are prefixed with the word **set**. For example, if there are two sets defined as:

```
set S1 mol. 1 (molecule number 1)  
set S2 atom.ptype C,O,N,CA (all backbone atoms)
```

the following are then valid atom specifications:

- **set S1 and set S2**, all backbone atoms in molecule 1
- **set S1 or set S2**, all backbone atoms or atoms in molecule 1
- **within 5.0 set S1**, all atoms within 5.0 Å of molecule 1.

## 3.8 Special Specifications

- The specification **all** matches everything. You can use this in any Maestro command which expects an ASL operand in order to apply the command to all the atoms. For example to color all atoms green:

```
coloratom color=green all
```

Note that the syntax for this is just **all**, not **atom. all** or **molecule. all**.

- Anything enclosed in / / is treated as a string in the linear substructure notation - see the BatchMin Reference Manual for a complete description of this. Some examples are:

```
/C3(-H1)(-H1)(-H1)/ specifies all methyl groups  
/C2(=O2)-N2-H2/ specifies all amide groups  
/C0-N0/ specifies all pairs bound C-N atoms, and  
/O0-S0/ specifies any atom bound to a sulfur.
```

Specifications made in this way can be treated in the same way as any other specification in the ASL, e.g. they can have operators applied to them.

## 3.9 Matching PDB Atom Names

Because PDB atom names are four characters wide we need to employ the following strategy in order to conveniently specify PDB atom names:

- Unquoted names which begin with a non-numeric character have a blank character inserted in front of them and are padded with blanks from the right to make up four characters before matching. Examples:
  - Property List: CA,C  
Actually Matched: " CA "," C "
  - Property List: CG1,CG2  
Actually Matched: " CG1 "," CG2"
- Unquoted names which begin with a number do not have an initial blank character inserted but are right to padded four characters. Example:
  - Property List: CA,1HB,2HB  
Actually Matched: " CA ","1HB ","2HB "
- Quoted names (double or single quotes are acceptable) are treated as literals but will be right padded with blanks to make up four characters. Examples:
  - Property List: CA,"CA"  
Actually Matched: " CA ","CA " (alpha carbons or calciums)
  - Property List: " N A"  
Actually Matched: " N A" (heme atoms)

## 3.10 Miscellaneous

### 3.10.1 Atoms not yet present in a structure

If a structure in the Workspace has only 100 atoms when the ASL definition: `atom.num 1,8,44,101,103` is issued, Maestro will simply match the atoms numbered 1,8 and 44. If additional atoms are subsequently added to the structure, the atoms bearing the numbers 101 and 103 will be added to the previously defined set.

### 3.10.2 Aliasing

Maestro allows you to define your own aliases, using either the Command Input Area or the Command Aliases panel. Maestro converts all aliases into their corresponding commands before performing operations involving the

aliased commands. Users must ensure that aliases produce sensible results. Some aliases are supplied with the distribution. They are:

Operator: **and**

Aliases: intersection, INTERSECTION, &

Operator: **or**

Aliases: UNION, union, |

Operator: **not**

Aliases: !

Class Designator: **mol.**

Aliases: MOL, mol

Class Designator: **atom.**

Aliases: ATOM, atom

Class Designator: **res.**

Aliases: RES, res

Class Designator: **chain.**

Aliases: CHAIN, chain

ASL Definition: **atom. ca,c,n,h,o**

Aliases: BACKBONE, backbone

ASL Definition: **not (atom.pt ca,c,n,h,o)**

Aliases: SIDECHAIN, sidechain

ASL Definition: **"/H2-O3-H2/ or atom.mtype OW"**

Aliases: WATER, water

### **3.11 Useful Hints when using ASL with the Project Facility**

Maestro 4.1 introduced its Project Facility. Entries can be included into and excluded from the Workspace. The order in which this is performed affects the molecule numbers. For example, if you have two entries in your Project Table called “A” and “B” and you include into an empty Workspace first A and then B, the molecule numbers will be 1 for A and 2 for B; however, if you first include B and then A, the molecule numbers will be 1 for B and 2 for A.

This means that an expression such as **mol. 1** will match different atoms in each of the above cases. In earlier versions of Maestro, it often made sense to use the **mol.** notation because there was no Project Facility. But with the Project Facility, Maestro is entry-centric and in most cases it makes more sense to use entry names.

For example, if you have an inhibitor and a receptor that are in different entries and wish to have a ribbon appear on only the receptor, use the entry name in the ASL expression, not the molecule number. This will ensure that when the receptor is included that it, and only it, will be used to generate the ribbon(s). A different inclusion order of entries in the Workspace will then result in the same matching atoms. So for ribbons with a receptor called "receptor" it is more useful to use `entry.name receptor` as the ASL definition.

## 3.12 ASL Examples

This section gives some examples of the use of the ASL in real-life situations. Note that while these examples all use lower-case, the ASL expressions themselves are not case sensitive.

i) Defining a set to refer to a ligand and/or receptor.

The exact command will depend on the nature of your system. If the ligand and the receptor are separate entries then it will suffice to use

```
set ligand entry.name <ligand_name>
```

where `<ligand_name>` is the name of the entry that contains the ligand. Similarly

```
set receptor entry.name <receptor_name>
```

for the receptor with entry called `<receptor_name>`.

In order to define sets that will work with multiple ligands it's also possible to define the ligand as everything that is not part of the receptor. A definition of:

```
set ligand not set receptor
```

will identify the ligand as anything that's not part of the receptor.

If the ligand and the receptor are part of the same entry then molecule numbers are the best way to define the ligand and the receptor. Assuming the receptor is molecule 1 and the ligand molecule 2:

```
set ligand mol.num 2  
set receptor mol.num 1
```

Note however that the use of molecule numbers in set definitions should be avoided where possible as these depend on the order in which the project entries are included into the Workspace. If it is possible to use entry names, then these should be used.

The subsequent examples assume that sets for the receptor and the ligand have been defined using one of the methods defined above.

ii) The set of atoms within a given distance of the ligand.

One common task is to do something with the set of atoms within a given distance of the ligand. For example to only display those atoms or to include them in a substructure region for a MacroModel calculation. These examples will use the "displayonlyatom" command but the ASL which follows can be used with any other command that uses ASL.

To only display atoms within 5.0 Angstroms of the ligand:

```
displayonlyatom within 5.0 set ligand
```

A common variation is to display complete residues which have any of their atoms within a given distance of the ligand:

```
displayonlyatom fillres within 5.0 set ligand
```

It's also possible to restrict the expression so that it only applies to receptor atoms within a given distance of the ligand. Here the Boolean "and" operator is used to restrict the displayed atoms to the receptor only:

```
displayonlyatom set receptor and fillres within 5.0 set  
ligand
```

Because this is a lengthy expression it's often convenient to make this into a set itself:

```
set active_site set receptor and fillres within 5.0 set  
ligand
```

An equivalent form of this is:

```
set active_site (! set ligand) & fillres within 5.0 set  
ligand
```

Note that "!" is a standard alias for "not" and similarly "&" for "and".

iii) Sidechain and backbone.

The ASL has standard aliases for the definition of sidechain and backbone atoms in proteins. For example to only display the atoms of the backbone:

```
displayonlyatom backbone
```

These aliases can be used with operators to build up more complicated expressions. For example to only display the sidechain of the receptor:

```
displayonlyatom sidechain and set receptor
```

To display only the sidechains of the atoms within 5.0 Angstroms of the ligand:

```
displayonlyatom sidechain and set receptor and fillres  
within 5.0 set ligand
```

iv) Atoms of a given type.

There are a variety of ways to specify atoms of a given type. For example to specify all carbons, nitrogens and oxygens the following is used:

```
atom.ele C,N,O
```

To specify non-hydrogen atoms:

```
not atom.ele H
```

To specify the alpha carbons in a protein:

```
atom.ptype CA
```

To specify all sp<sub>2</sub> carbons there are two choices. The first relies on knowing that the MacroModel atom type for such an atom is "C2" and using:

```
atom.mtype C2
```

The other (assuming no formally charged or radical carbons are present) uses the number of attachments to the atom:

```
atom.ele C and atom.att 3
```

To specify polar hydrogens:

```
atom.ele H and not /CO-HO/
```

or

```
atom.ele H and not atom.mtype H1
```

v) Water molecules.

The ASL has a standard alias "water". For example to delete all water molecules the Maestro command is:

```
delete atom water
```

vii) Restricting an operation to the atoms that are currently displayed in the workspace.

Often a user will be working with only a subset of the atoms in the workspace displayed. If an operation is to be performed only on the atoms that are displayed then the "atom.displayed" property can be used. For example to change the color to green of all the atoms currently displayed in the Workspace and to leave alone the undisplayed Workspace atoms:

```
coloratom color=green atom.disp
```

To only do it for the atoms that are displayed and in the receptor:

```
coloratom color=green atom.disp and receptor
```

viii) Specifying molecules.

All molecules with between 30 and 100 atoms:

```
mol.atoms 30-100
```

All molecules with over 100 atoms:

```
mol.atoms >100
```

All molecules with a molecular weight over 300:

```
mol.weight > 300.0
```

All molecules which contain a halogen:

```
fillmol atom.ele F,Cl,Br,I
```

ix) Specifying atoms based on a linear-substructure notation.

The ASL supports the use of a SMILES-like linear substructure notation to specify atoms with a particular bonding arrangement. The atoms are referred to by MacroModel atom types, but there are wildcard types that

can be used to allow the expression to apply to any atoms of a given element type.

Some examples:

Any five-membered ring:

```
/00-00-00-00-00-1/
```

Aromatic six-membered carbon rings(C2 is sp<sup>2</sup> carbon)

```
/C2-C2*C2-C2*C2-C2*1/
```

Amide groups:

```
/C2(*O2)-N2/
```

Methyl groups:

```
/C3(-H1)(-H1)(-H1)/
```

Water:

```
/H2-O3-H2/
```

Guanadinium group:

```
/N2(-H3)-C2(*N4(-H4)(-H4))-N2(-H3)(-H3)/
```

x) Using wildcard characters.

Most string-type property values can use wildcard characters. Some examples:

All PDB atom names beginning with C

```
atom.ptype C*
```

All forms of the histidine residue:

```
res.ptype HI*
```

All entries that begin with "lig":

```
entry.name lig*
```

xi) Addressing residues with negative residue numbers.

From version 6.1 of Maestro it is possible to use residue numbers that are negative or zero.

```
res. -8
```

will match residues with residue number -8.

xii) Use SMARTS expressions.

From version 7.0 of Maestro it is possible to use SMARTS expressions as part of an ASL expression.

Some examples:

```
smarts. CCC
```

will match all three-carbon subsequences

```
smarts. [R] and atom.ele N
```

will match all ring nitrogens

```
smarts. C1CCCCC1
```

will match all six-membered carbon rings

## 4 The Entry Specification Language

This chapter documents the syntax of Maestro's Entry Specification Language (ESL).

### 4.1 Why an Entry Selection Language?

The Entry Selection Language (ESL) is used for selecting entries in a Maestro project based on the properties of those entries. ESL expressions can be of arbitrary complexity and can use parentheses and logical operators. While the ESL is similar in appearance to the Atom Specification Language (ASL), they are used for quite different purposes - the ASL for specifying sets of atoms in the workspace, the ESL for selecting entries in the project. A typical application for the ESL would be for filtering. An ESL expression can be used to select only those entries that meet a particular criteria, usually based on entry properties. Once selected, those entries can be displayed in the workspace or exported to an external file.

### 4.2 Entry Properties

There are a number of properties that can be used in ESL expressions:

**entry**      The name of the entry is the most useful property at present.  
The syntax is:

```
entry <name>
entry_re <name_expression>
```

Entry names that contain spaces need to be enclosed in double quotes. The following wildcard characters can be used for the **entry\_re** expression:

- \* matches any number of any characters
- ? matches any single character
- # matches any single digit

Examples:

```
entry anentry
entry crambin-1
entry_re cramb*
entry_re ligand##
```

Note: **name** can be used as an alias for **entry** and **name\_re** can be used as an alias for **entry\_re**.

**selected** This allows the selection of entries based on whether or not they are already selected in the project. Examples:

```
entry_re lig* and not selected  
not selected
```

**included** This allows the selection of entries based on whether or not they are already included for display in the workspace.

```
entry_re lig* and not included  
not included
```

**all** This is a quick way to select all entries.

### 4.3 Logical Operators

The following logical operators are supported (in order of decreasing priority)

- **not**
- **and**   **or**

At equal levels of priority expression will be evaluated from left to right. Parentheses can be used to override the default order of priority.

### 4.4 Entry Property Comparisons

ESL expressions may also contain comparisons involving entry properties. Properties may be defined by their data names or their user names. The data names are the names that appear in the Maestro files. An example is “i\_qp\_n\_stars”. User names are the names that appear in the first row of Maestro’s Project Table. An example of how this would appear in Maestro’s Project Table Header is **n stars**. User names are specified in commands as:

```
user(< propertyname >)
```

For example:

```
user(volume)
```

and so,

```
user(n stars) and  
i_qp_n_stars
```

are equivalent names for properties.

For integer and real type properties the following comparison operators are supported: ==, !=, <, >, <=, >=.

For boolean (logical) properties, the name of the property can just be specified and will match any entries where that property is true. It is also possible to use == and != in explicit comparisons with the values **true** and **false**.

Comparisons involving string properties have the form:

⟨property name⟩ ⟨string literal⟩

For example:

```
s_m_string_prop left
s_m_string_prop left*
```

String literals may include the \*, ?, and # wildcard characters available for entry names as described above.

## 4.5 Examples

Some examples of the use of the ESL are:

```
entry anentry
entry_re ligand*
entry_re lig* and selected
(entry_re lig# and selected ) and not included
user(volume) < 100.0 and selected
included and user(n stars) == 2 and user(n amine) == 0
user(log s) > 0.5 and user(log S) < 1.0
```



## 5 Commands

### **1ddataset**

Controls the appearance of the a single dataset on a plot of a dihedral drive of a single dihedral.

Syntax:

```
1ddataset ccolor=black | red | green | blue | purple | orange |
    | blue_green | light_green | red_purple | yellow | cyan
    | curve=solid | dashed | noline cwidth=⟨n⟩ scolor=black | red
    | green | blue | purple | orange | blue_green | light_green |
    | red_purple | yellow | cyan ssize=⟨n⟩ symbol=filled_rectangle |
    hollow_rectangle | filled_circle | hollow_circle | cross | point |
    filled_diamond | hollow_diamond | no_symbol ⟨data_set_name⟩
    | [⟨grd_file_name⟩]
```

Options:

*ccolor* The color of the curve used on the graph for this dataset.

Valid values: black  
red  
green  
blue  
purple  
orange  
blue\_green  
light\_green  
red\_purple  
yellow  
cyan

Default value: **black**

*curve* The type of curve which will join the points on the graph.

Valid values: solid  
dashed  
noline

Default value: **solid**

*cwidth* The width of the line used on the graph for this dataset.

Valid values: integers

Default value: **1**  
Minimum: **1**  
Maximum: **5**

*scolor* The color of the symbols used on the graph for this dataset.

Valid values: black  
red  
green  
blue  
purple  
orange  
blue\_green  
light\_green  
red\_purple  
yellow  
cyan

Default value: **black**

*ssize* The size of the symbol used on this plot for this dataset.

Valid values: integers  
Default value: **3**  
Minimum: **2**  
Maximum: **5**

*symbol* The type of symbol used for this data set on the graph.

Valid values: filled\_rectangle  
hollow\_rectangle  
filled\_circle  
hollow\_circle  
cross  
point  
filled\_diamond  
hollow\_diamond  
no\_symbol

Default value: **filled\_rectangle**

Operands:

$\langle \text{data\_set\_name} \rangle$  [ $\langle \text{grd\_file\_name} \rangle$ ]

The first operand is the name of the dataset which is to be created or to have its properties changed. If a second operand is present then that is treated as the name of a file from which the data is to be read. The full name of the file, including the suffix, should be included.

## 1dplot

Controls the appearance of the plot the energy as a function of a drive of a single dihedral angle.

Syntax:

```
1dplot amax=<x> amin=<x> emax=<x> emin=<x>
          escale=absolute | relative orientation=landscape | portrait
          papersize=letter | a4 squareplot=yes | no units=kj | kcal
          [postscript_file_name]
```

Options:

*amax* The maximum angle value displayed on the graph

Valid values:    real

Default value: **0**

*amin* The minimum angle value displayed on the graph

Valid values:    real

Default value: **0**

*emax* The maximum energy value displayed on the graph

Valid values:    real

Default value: **0**

*emin* The minimum energy value displayed on the graph

Valid values:    real

Default value: **0**

*escale* The energy scale (absolute/relative) for the graph

Valid values:    absolute

                  relative

Default value: **absolute**

*orientation*

The paper orientation for the postscript output

Valid values:    landscape

                  portrait

Default value: **landscape**

*papersize* The paper size for the postscript output

Valid values:    letter

                  a4

Default value: **letter**

*squareplot* A boolean which controls whether the plot will be constrained to be square.

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **false**

*units* The energy units for the graph

Valid values: kj  
                  kcal  
Default value: **kJ**

Operands:

[⟨ postscript\_file\_name ⟩]

If an operand is given then this will be treated as the name of a file to which a postscript representation of the plot will be written. The full name of the file, including any suffix, must be included.

## **1drescale**

Rescales the 1D plot axes back to be able to view the entire data range.

Syntax:

### **1drescale**

## **1dtable**

Write out a table of the data from the 1D plot to an external file.

Syntax:

### **1dtable**

## **2ddataset**

Controls the appearance of the dataset plotted on the 2D grid contour plot.

Syntax:

**2ddataset** *cwidth=⟨n⟩ emax=⟨x⟩ emin=⟨x⟩ negatedashed=yes | no numcontours=⟨n⟩*

Options:

*cwidth* The width of the contours drawn on the plot

Valid values: integers

Default value: **2**

Minimum: 1

Maximum: 5

*emax* The maximum energy value displayed on the graph

Valid values: reals

Default value: **0**

*emin* The minimum energy value displayed on the graph

Valid values: reals

Default value: **0**

*negatedashed*

A boolean which controls whether contours corresponding to negative values will be drawn with a dashed line.

Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **false**

*numcontours*

The number of contours used on the graph

Valid values: integers

Default value: **10**

Minimum: 2

Maximum: 10

## 2dplot

Controls the appearance of a plot of the energy as a function of two dihedral angles.

Syntax:

**2dplot** *escale=absolute | relative orientation=landscape | portrait  
papersize=letter | a4 squareplot=yes | no units=kj | kcal  
[⟨ postscript\_file\_name ⟩]*

Options:

*escale* The energy scale (absolute/relative) for the graph

Valid values: absolute

relative

Default value: **absolute**

*orientation*

The paper orientation for the postscript output

Valid values: landscape

portrait

Default value: **landscape**

*papersize* The paper size for the postscript output

Valid values: letter

a4

Default value: **letter**

*squareplot* A boolean which controls whether the plot will be constrained to be square.

Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **false**

*units* The energy units for the graph

Valid values: kj

kcal

Default value: **kj**

Operands:

[⟨ postscript\_file\_name ⟩]

If an operand is given, this is treated as the name of a file to which a postscript representation of the file is to be written.

## 2drescale

Rescales the plot axes back to be able to view the entire data range.

Syntax:

## **2drescale**

### **addfromhold**

This command retrieves the structure from the hold set and adds it to the current on-screen structure. The added hold structure is placed at the periphery of the on-screen structure and then a scale-to-screen is done.

Syntax:

**addfromhold** <hold\_name>

Operands:

<hold\_name>

The name of the hold. This must be the name which was specified when the hold was created using the “hold” command.

### **adjustangle**

Adjust the bond angle specified by the 3 atoms to the given value.

Syntax:

**adjustangle** *angle*=<x> *move*=attached | terminal | single  
<atom1> <atom2> <atom3>

Options:

*angle* Value to which to set the angle

Valid values: reals

Default value: **0**

Minimum: 0.0

Maximum: 180.0

*move* This option sets how to move other atoms that attached to the selected moving atom. There are three options: (1) move all attached atoms, (2) move all attached terminal atoms, and (3) move only single atom. Default option is (1).

Valid values: attached

terminal

single

Default value: **attached**

Operands:

$\langle \text{atom1} \rangle \langle \text{atom2} \rangle \langle \text{atom3} \rangle$

Three atoms which are used to adjust an angle. Note that direction does matter. So a-b-c is different from c-b-a. The third atom 'points' to the part of the structure that will be adjusted.

## adjustdihedral

Adjust the dihedral specified by 4 atoms to the given value.

Syntax:

**adjustdihedral** *dihedral*= $\langle x \rangle$  *dihedralterm*= $\langle x \rangle$  *move*=**attached** |  
terminal | single  $\langle \text{atom1} \rangle \langle \text{atom2} \rangle \langle \text{atom3} \rangle \langle \text{atom4} \rangle$

Options:

*dihedral* Value to which to set the torsion

Valid values: reals

Default value: **0**

Minimum: -180.1

Maximum: 180.1

*dihedralterm*

Value to which to terminally attached to the dihedral

Valid values: reals

Default value: **0**

Minimum: -180.1

Maximum: 180.1

*move*

This option sets how to move other atoms that attached to the selected moving atom. There are three options: (1) move all attached atoms, (2) move all attached terminal atoms, and (3) move only single atom. Default option is (1).

Valid values: attached

terminal

single

Default value: **attached**

Operands:

$\langle \text{atom1} \rangle \langle \text{atom2} \rangle \langle \text{atom3} \rangle \langle \text{atom4} \rangle$

Four atoms which are used to adjust a torsion. Note that direction does matter. So a-b-c-d is different from d-c-b-a. The fourth atom 'points' to the part of the structure that will be adjusted.

## adjustdistance

Adjust the distance specified by 2 atoms to the given value.

Syntax:

**adjustdistance**  $distance=\langle x \rangle$   $move=$ attached | terminal | single  
 $\langle \text{atom1} \rangle \langle \text{atom2} \rangle$

Options:

*distance* Value to which to set the torsion

Valid values: reals

Default value: 0

Minimum: 0.0

*move* This option sets how to move other atoms that attached to the selected moving atom. There are three options: (1) move all attached atoms, (2) move all attached terminal atoms, and (3) move only single atom. Default option is (1).

Valid values: attached

terminal

single

Default value: **attached**

Operands:

$\langle \text{atom1} \rangle \langle \text{atom2} \rangle$

Two atoms which are used to adjust a distance. Note that direction does matter. So a-b is different from b-a. The second atom 'points' to the part of the structure that will be adjusted.

## **alias**

Define an alias for a command. This allows an abbreviated symbol to be defined for any command.

Syntax:

**alias** <alias\_name> <definition>

Operands:

<alias\_name> <definition>

The first operand is the name of the alias. If this contains embedded spaces then it must be enclosed in double quotes. The remaining operands are the definition of the alias. When command processing takes place, all occurrences of the alias name (the first operand) will be replaced by the definition of the alias.

## **angle**

Specifies a triplet of atoms to have their bond angle measured and displayed.

Syntax:

**angle** <atom1> <atom2> <atom3>

Operands:

<atom1> <atom2> <atom3>

The three atoms between which the angle is to be measured. Note that specifying a-b-c is the same as specifying c-b-a

## **appendribbons**

Generate a ribbon for the parts that are not currently in ribbon.

Syntax:

## **appendribbons**

### **application**

Determines which backend application is currently active and which application-specific menu is displayed. For example the command application Impact will result in the Impact menu being displayed in the main menu bar.

Syntax:

**application** <application\_name>

Operands:

<application\_name>

The name of the backend application which Maestro is currently interacting with. Current options are “none”, “macromodel” or “impact”.

### **atom**

Create a new atom in space with the current type and at the position specified by the operands.

Syntax:

**atom** *by=element | type element=<text> type=<n> <x> <y> <z>*

Options:

*by* This option determines whether atoms are to be described by element symbols (the element= option) or by atom types (type= option).

Valid values: element

type

Default value: **element**

*element* This option sets the current element of all atoms to be created (or retyped) if the by option == element

Valid values: text strings

Default value: **C**

*type*      This option sets the current type of all atoms to be created (or retyped) if the by option == type  
Valid values:    integers  
Default value:    **3**

Operands:

$\langle x \rangle \langle y \rangle \langle z \rangle$

The operands are three real numbers which specify the x, y and z coordinates for the new atom.

## **atomname**

Set the PDB atom name for all atoms which match the ASL specification.

Syntax:

**atomname** <PDBNAME> <ASL>

Operands:

$\langle \text{PDBNAME} \rangle \langle \text{ASL} \rangle$

The first operand is the PDB atom name which will be used for all atoms which match the specification. The second operand is a valid ASL string which defines the set of atoms which are to have their atom names changed.

## **attach**

Attach the currently selected fragment to the specified atom. The bond to this atom (and the atom itself) will be replaced by a bond from the incoming fragment.

Syntax:

**attach** <atom\_num>

Operands:

$\langle \text{atom\_num} \rangle$

The atom number defines the bond to which the fragment be attached. The atom specified must be a terminally attached atom.

## **attachmentmarkerdump**

Print out the current option values of the attachment marker command.

Syntax:

## **attachmentmarkerdump**

## **attachmentmarkersettings**

Set graphical data of attachment markers.

Syntax:

```
attachmentmarkersettings ambient=⟨x⟩ blue=⟨x⟩  
    cornradius=⟨x⟩ cylinderheight=⟨x⟩ cylindrerradius=⟨x⟩  
    diffuse=⟨x⟩ drawstyle=solid | line emission=⟨x⟩ green=⟨x⟩  
    linewidth=⟨n⟩ reagentradius=⟨x⟩ red=⟨x⟩ selectblue=⟨x⟩  
    selectgreen=⟨x⟩ selectedred=⟨x⟩ shininess=⟨x⟩ sliceline=⟨n⟩  
    slicesolid=⟨n⟩ specular=⟨x⟩ stackline=⟨n⟩ stacksolid=⟨n⟩  
    transparency=⟨x⟩
```

Options:

*ambient* Set material property - ambient, to its red, green, and blue components, for front face.  
Valid values: reals  
Default value: **0.5**  
Minimum: 0.0  
Maximum: 1.0

*blue* The blue component of attachment markers.  
Valid values: reals  
Default value: **0**  
Minimum: 0.0  
Maximum: 1.0

*cornradius* The radius of corn of attachment markers.  
Valid values: reals  
Default value: **0.55**  
Minimum: 0.0

*cylinderheight*

The cylinder height ratio of attachment markers.

Valid values:   reals

Default value: **0.6**

Minimum:       0.0

Maximum:      1.0

*cylinderradius*

The radius of cylinder of attachment markers.

Valid values:   reals

Default value: **0.26**

Minimum:       0.0

*diffuse*

Set material property - diffuse, to its red, green, and blue components, for front face.

Valid values:   reals

Default value: **0.4**

Minimum:       0.0

Maximum:      1.0

*drawstyle*

The styles of rendering attachment markers, they are: 1 - solid, and 2 - lines. Default is solid.

Valid values:   solid

line

Default value: **solid**

*emission*

Set material property - emission, to its red, green, and blue components, for front face.

Valid values:   reals

Default value: **0.05**

Minimum:       0.0

Maximum:      1.0

*green*

The green component of attachment markers.

Valid values:   reals

Default value: **0.6**

Minimum:       0.0

Maximum:      1.0

*linewidth*

Set the width of lines in drawing attachment.

Valid values:   integers

Default value: **2**

Minimum:       1

*reagentradius*

The radius of sphere for attachment markers having associated reagents.

	Valid values:	reals
	Default value:	<b>0.8</b>
	Minimum:	0.0
<i>red</i>	The red component of attachment markers.	
	Valid values:	reals
	Default value:	<b>1</b>
	Minimum:	0.0
	Maximum:	1.0
<i>selectblue</i>	The blue component of selected attachment markers.	
	Valid values:	reals
	Default value:	<b>1</b>
	Minimum:	0.0
	Maximum:	1.0
<i>selectgreen</i>	The green component of selected attachment markers.	
	Valid values:	reals
	Default value:	<b>0.9</b>
	Minimum:	0.0
	Maximum:	1.0
<i>selectedred</i>	The red component of selected attachment markers.	
	Valid values:	reals
	Default value:	<b>0.2</b>
	Minimum:	0.0
	Maximum:	1.0
<i>shininess</i>	Set material property - shininess, for front face.	
	Valid values:	reals
	Default value:	<b>80</b>
	Minimum:	0.0
	Maximum:	128.0
<i>sliceline</i>	Set the slices of drawing line attachment.	
	Valid values:	integers
	Default value:	<b>10</b>
	Minimum:	2
<i>slicesolid</i>	Set the slices of drawing solid attachment.	
	Valid values:	integers
	Default value:	<b>36</b>
	Minimum:	2
<i>specular</i>	Set material property - specular, to its red, green, and blue components, for front face.	

	Valid values:	reals
	Default value:	<b>0.2</b>
	Minimum:	0.0
	Maximum:	1.0
<i>stackline</i>	Set the stacks of drawing line attachment.	
	Valid values:	integers
	Default value:	<b>8</b>
	Minimum:	2
<i>stacksolid</i>	Set the stacks of drawing solid attachment.	
	Valid values:	integers
	Default value:	<b>18</b>
	Minimum:	2
<i>transparency</i>	The transparency of rendering attachment markers.	
	Valid values:	reals
	Default value:	<b>20</b>
	Minimum:	0.0
	Maximum:	100.0

## autosec

Performs the automatic setup for the conformational search. The setup is only performed on those atoms which match the ASL definition given in the operand.

Syntax:

```
autosec chiralatoms=yes | no compatoms=yes | no
rings=yes | no torsionchecks=yes | no <ASL>
```

Options:

### *chiralatoms*

A boolean option which determines whether the the automatic setup will include finding chiral atoms.

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **true**

### *comptatoms*

A boolean option which determines whether the the automatic setup will include finding comparison atoms.

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **true**

*rings* A boolean option which determines whether the the automatic setup will include finding ring closures.

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **true**

*torsionchecks*

A boolean option which determines whether the the automatic setup will include finding torsion angles to be checked.

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **true**

Operands:

*<ASL>*

A string in the atom specification language. The automatic setup will be peformed for only that atoms which match this ASL specification.

## **beginundoblock**

Begin a new undoable command block.

Syntax:

**beginundoblock**

## **bmincomfile**

Write a MacroModel command file with the current energy settings.

Syntax:

**bmincomfile**

## **bond**

Create a new bond between two atoms or increments the bond order between two existing atoms.

Syntax:

**bond** <atom1> <atom2>

Operands:

<atom1> <atom2>

The operands are the numbers of the two atoms which are to be connected by a bond or have the bond order of an existing bond incremented.

## **bondorder**

Increment, decrement or set the bond order for the specified bond.

Syntax:

**bondorder** increment|decrement|<bond\_order> <atom1> <atom2>

Operands:

increment|decrement|<bond\_order> <atom1> <atom2>

The first operand is either the word “increment”, the word “decrement”, or an integer value representing the bond order to be applied to the specified bond. Note that “0” is a valid bond order. The second and third operands are the atoms which define the bond which is to have its bond order changed.

## **bondtonew**

Create a new atom and bond it with a single bond to an existing atom.

Syntax:

**bondtonew** <atom\_num> <x> <y> <z>

Operands:

<atom\_num> <x> <y> <z>

The first operand is the number of an existing atom from which the bond is to be drawn. The remaining three operands are the x, y and z coordinates at which the new atom is to be placed.

## buildoptions

Sets options associated with structure building. These are persistent and will be retained between Maestro sessions.

Syntax:

```
buildoptions adjustbondlengths=yes | no  
                 adjustnumhydrogens=yes | no useunitedatoms=yes | no
```

Options:

### *adjustbondlengths*

When an atom type is changed, bond lengths around that atom will be set to ideal values.

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **true**

### *adjustnumhydrogens*

During bond order, drawing or formal charge change operations, rectify the number of hydrogens to be consistent with the new bonding or formal charge.

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **true**

### *useunitedatoms*

Will allow the generation of united atom types while structure building.

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **true**

## calcenergy

Used to set options associated with a single point energy calculation.

Syntax:

```
calcenergy listing=none | minimal | complete
```

Options:

*listing* This option determines the extent of the listing of energy components.

Valid values:      none  
                      minimal  
                      complete  
Default value:     **none**

Aliases:

**ecalc** (see [ecalc], page 95)

## **canonicalname**

Sets atom names to one generated from atom properties (such as element or PDB names) for all the atoms matched by the ASL operand.

Syntax:

**canonicalname** <separator> <option\_flag> <ASL>

Operands:

<separator> <option\_flag> <ASL>

The first operand is the separator symbol to be used in generating the canonical names of all atoms. The second operand is the option\_flag, which indicates which two atom properties will be used to construct the canonical names. Possible values are “3” - from element and atom number and “12” PDB chain, residue and atom name. The final operand is the ASL expression to indicate which atoms are to have a canonical name set.

## **cascadepanels**

Cascade visible panels

Syntax:

**cascadepanels**

## **cd**

This is a standard alias for **changedirectory** (see [changedirectory], page 57).

## **cellsmarkerdump**

Print out the current option values of the Glide hydrophobic Cells markers command.

Syntax:

**cellsmarkerdump**

## **cellsmarkersettings**

Set graphical data of Glide hydrophobic cells markers.

Syntax:

```
cellsmarkersettings activeblue=<x> activegreen=<x>
    activered=<x> ambient=<x> blue=<x> diffuse=<x>
    emission=<x> green=<x> highlightblue=<x>
    highlightgreen=<x> highlightred=<x> labelregions=yes | no
    red=<x> regionblue=<x> regiongreen=<x> regionred=<x>
    roundingeffect=<x> shininess=<x> specular=<x> step=<n>
    transparency=<x>
```

Options:

*activeblue* The blue color component of cell active frame and region label.

Valid values:    reals  
 Default value: **0**  
 Minimum:        0.0  
 Maximum:        1.0

*activegreen*

The green color component of cell active frame and region label.

Valid values:    reals  
 Default value: **1**  
 Minimum:        0.0  
 Maximum:        1.0

*activered* The red color component of cell active frame and region label.

Valid values:    reals  
 Default value: **1**  
 Minimum:        0.0  
 Maximum:        1.0

<i>ambient</i>	Set material property - ambient, to its red, green, and blue components, for front face.
	Valid values:    reals
	Default value: <b>0.5</b>
	Minimum:    0.0
	Maximum:    1.0
<i>blue</i>	The blue color component of cell markers.
	Valid values:    reals
	Default value: <b>0.75</b>
	Minimum:    0.0
	Maximum:    1.0
<i>diffuse</i>	Set material property - diffuse, to its red, green, and blue components, for front face.
	Valid values:    reals
	Default value: <b>0.5</b>
	Minimum:    0.0
	Maximum:    1.0
<i>emission</i>	Set material property - emission, to its red, green, and blue components, for front face.
	Valid values:    reals
	Default value: <b>0.1</b>
	Minimum:    0.0
	Maximum:    1.0
<i>green</i>	The green color component of cell markers.
	Valid values:    reals
	Default value: <b>0.75</b>
	Minimum:    0.0
	Maximum:    1.0
<i>highlightblue</i>	The blue color component of cell markers if the cell is of a highlight region.
	Valid values:    reals
	Default value: <b>0.3</b>
	Minimum:    0.0
	Maximum:    1.0
<i>highlightgreen</i>	The green color component of cell markers if the cell is of a highlight region.
	Valid values:    reals
	Default value: <b>0.3</b>

Minimum: 0.0  
 Maximum: 1.0

*highlightred*

The red color component of cell markers if the cell is of a highlight region.

Valid values: reals  
 Default value: **1**  
 Minimum: 0.0  
 Maximum: 1.0

*labelregions*

This option determines whether a region should be labeled (true) or not (false).

Valid values: boolean (true|false; yes|no; y|n; on|off)  
 Default value: **true**

*red*

The red color component of cell markers.

Valid values: reals  
 Default value: **0.75**  
 Minimum: 0.0  
 Maximum: 1.0

*regionblue*

The blue color component of cell markers if the cell is of a region.

Valid values: reals  
 Default value: **0**  
 Minimum: 0.0  
 Maximum: 1.0

*regionongreen*

The green color component of cell markers if the cell is of a region.

Valid values: reals  
 Default value: **0**  
 Minimum: 0.0  
 Maximum: 1.0

*regionred*

The red color component of cell markers if the cell is of a region.

Valid values: reals  
 Default value: **1**  
 Minimum: 0.0  
 Maximum: 1.0

*roundingeffect*

This determines the rounding effect of edges of a cell, for front face.

Valid values: reals

	Default value:	<b>8</b>
	Minimum:	0.0
<i>shininess</i>	Set material property - shininess, for front face.	
	Valid values:	reals
	Default value:	<b>80</b>
	Minimum:	0.0
	Maximum:	128.0
<i>specular</i>	Set material property - specular, to its red, green, and blue components, for front face.	
	Valid values:	reals
	Default value:	<b>0.1</b>
	Minimum:	0.0
	Maximum:	1.0
<i>step</i>	The step domain tolerance of cells.	
	Valid values:	integers
	Default value:	<b>3</b>
	Minimum:	1
<i>transparency</i>	The transparency of phobic cell markers.	
	Valid values:	reals
	Default value:	<b>30</b>
	Minimum:	0.0
	Maximum:	100.0

## centeratom

Set global center of rotation to the given atom

Syntax:

**centeratom** *atom*=⟨n⟩

Options:

<i>atom</i>	Atom to center transformations on
	Valid values:
	Default value:
	Minimum:

## **centerbond**

Set global center of rotation to the given bond

Syntax:

**centerbond** *at1=⟨ n ⟩ at2=⟨ n ⟩*

Options:

*at1* Atom1 of bond to be center of transformation

Valid values: integers

Default value: **1**

Minimum: 1

*at2* Atom 2 of bond to be center of transformation

Valid values: integers

Default value: **1**

Minimum: 1

## **centercoordinates**

Set global center of rotation to the given coordinate

Syntax:

**centercoordinates** *x=⟨ x ⟩ y=⟨ x ⟩ z=⟨ x ⟩*

Options:

*x* X center

Valid values: reals

Default value: **0**

*y* Y center

Valid values: reals

Default value: **0**

*z* Z center

Valid values: reals

Default value: **0**

## **centroid**

This command takes the previously defined centroid atoms, averages the x, y, z co-ordinates and then creates a new atom of type 61 at the average position.

Syntax:

## **centroid**

## **centroidatom**

Define a single atom which will be used to define the centroid. Associated with this atom is a single atom marker.

Syntax:

## **centroidatom** $\langle \text{atom\_num} \rangle$

Operands:

$\langle \text{atom\_num} \rangle$

The number of the atom to be used to define the centroid.

## **cglidedockconstraintposition**

Specifies a constraint position in the receptor for a Glide calculation.

Syntax:

```
cglidedockconstraintposition feature= $\langle \text{n} \rangle$  index= $\langle \text{n} \rangle$   
    radius= $\langle \text{x} \rangle$  type= $\langle \text{n} \rangle$  use1=yes | no use2=yes | no  
    use3=yes | no use4=yes | no x= $\langle \text{x} \rangle$  y= $\langle \text{x} \rangle$  z= $\langle \text{x} \rangle$ 
```

Options:

*feature*      The constraint feature of position in docking.

Valid values:    integers

Default value:   **-1**

*index*      The index of position in docking.

	Valid values:	integers
	Default value:	<b>0</b>
<i>radius</i>	The radius of a Glide constraint position.	
	Valid values:	reals
	Default value:	<b>1</b>
	Minimum:	0.0001
<i>type</i>	The constraint type of position in docking.	
	Valid values:	integers
	Default value:	<b>0</b>
<i>use1</i>	The flag indicates if this position will be used in docking for group 1.	
	Valid values:	boolean (true false; yes no; y n; on off)
	Default value:	<b>false</b>
<i>use2</i>	The flag indicates if this position will be used in docking for group 2.	
	Valid values:	boolean (true false; yes no; y n; on off)
	Default value:	<b>false</b>
<i>use3</i>	The flag indicates if this position will be used in docking for group 3.	
	Valid values:	boolean (true false; yes no; y n; on off)
	Default value:	<b>false</b>
<i>use4</i>	The flag indicates if this position will be used in docking for group 4.	
	Valid values:	boolean (true false; yes no; y n; on off)
	Default value:	<b>false</b>
<i>x</i>	The X coordinate of a Glide constraint position.	
	Valid values:	reals
	Default value:	<b>0</b>
<i>y</i>	The Y coordinate of a Glide constraint position.	
	Valid values:	reals
	Default value:	<b>0</b>
<i>z</i>	The Z coordinate of a Glide constraint position.	
	Valid values:	reals
	Default value:	<b>0</b>

## cglidedockconstraintregion

Specifies a constraint region in the receptor for a Glide calculation.

Syntax:

```
cglidedockconstraintregion atoms=<n> feature=<n>
    index=<n> type=<n> use1=yes | no use2=yes | no
    use3=yes | no use4=yes | no <region_name>
```

Options:

*atoms*      The number of required ligand atoms of constraint region in docking.

Valid values:    integers  
Default value:    **1**

*feature*      The constraint feature of constraint region in docking.

Valid values:    integers  
Default value:    **-1**

*index*      The index of constraint region in docking.

Valid values:    integers  
Default value:    **0**

*type*      The constraint type of constraint region in docking.

Valid values:    integers  
Default value:    **0**

*use1*      The flag indicates if this constraint region will be used in docking for group 1.

Valid values:    boolean (true|false; yes|no; y|n; on|off)  
Default value:    **false**

*use2*      The flag indicates if this constraint region will be used in docking for group 2.

Valid values:    boolean (true|false; yes|no; y|n; on|off)  
Default value:    **false**

*use3*      The flag indicates if this constraint region will be used in docking for group 3.

Valid values:    boolean (true|false; yes|no; y|n; on|off)  
Default value:    **false**

*use4*      The flag indicates if this constraint region will be used in docking for group 4.

Valid values:    boolean (true|false; yes|no; y|n; on|off)  
Default value:    **false**

Operands:

`<region_name>`

The name which will be applied to the region. If the name contains embedded spaces then it must be enclosed in double quotation marks.

## **chainname**

Set the chain name for all atoms which match the ASL specification.

Syntax:

**chainname** <CHAINNAME> <ASL>

Operands:

<CHAINNAME> <ASL>

The first operand is the PDB chain name (a single character) that will be used for all atoms which match the specification. The second operand is a valid ASL string which defines the set of atoms which are to have their residue names changed.

## **changedirectory**

Change the current directory to that given by the operand of this command.

Syntax:

**changedirectory** <new\_directory>

Operands:

<new\_directory>

The name of the directory to be made the current directory.

Aliases:

**cd** (see [cd], page 48), **chdir** (see [chdir], page 57)

## **chdir**

This is a standard alias for **changedirectory** (see [changedirectory], page 57).

## **chiralatom**

Specifies an atom which is to be marked as “chiral” in the conformational search

Syntax:

**chiralatom** <atom\_number>

Operands:

<atom\_number>

The number of an atom which is to be added to the list of atoms which are to be considered as chiral in the conformational search.

## **clip**

Adjust the front and/or back clipping planes

Syntax:

**clip** *back*=<x> *front*=<x>

Options:

*back* Position at which to set the back clipping plane

Valid values: reals

Default value: **0**

*front* Position at which to set the front clipping plane

Valid values: reals

Default value: **0**

## **clusteratom**

Defines a single comparison atom to be used for XCluster jobs where clustering is based on Atomic RMS differences.

Syntax:

### **clusteratom** <atom>

Operands:

<atom>

The number of an atom which is to added to the list of comparison atoms for a clustering calculation.

### **clusterheavyatoms**

Specifies the atoms to be used in comparisons of conformers in XCluster jobs where clustering is based on Atomic RMS differences. The atoms specified are all of the heavy atoms and optionally hydrogens on oxygen and sulfur.

Syntax:

**clusterheavyatoms** *osh=yes | no*

Options:

*osh*      A boolean which controls whether hydrogens on oxygen and sulfur are included with heavy atoms.

Valid values:    boolean (true|false; yes|no; y|n; on|off)

Default value:    **false**

### **clusterset**

Specifies a set of atoms to be used in comparisons of conformers in XCluster jobs where clustering is based on Atomic RMS differences.

Syntax:

**clusterset** <ASL>

Operands:

<ASL>

A string in the atom specification language. All atoms which match this description will be added to the list of XCluster comparison atoms.

## **clustertorsion**

Specifies four atoms which define a torsion to be compared in XCluster jobs where clustering is based on Torsional RMS differences.

Syntax:

**clustertorsion** <atom1> <atom2> <atom3> <atom4>

Operands:

<atom1> <atom2> <atom3> <atom4>

The numbers of four atoms which define a torsion angle to be compared in XCluster jobs. Note that specifying a-b-c-d is the same as specifying d-c-b-a.

## **coloratom**

Set the color for a group of atoms defined by the ASL operand.

Syntax:

**coloratom** *cindex*=<n> *color*=<text> <ASL>

Options:

*cindex* An integer which indicates color index which is to be used for the atoms

Valid values: integers

Default value: **2**

Minimum: 1

Maximum: 256

*color* A string which is the color name for atom coloring. Valid color names are described in the file \$SCHRODINGER/maestro-vX.X/data/res/colors.res

Valid values: text strings

Default value:

Operands:

<ASL>

A string in the atom specification language. All atoms which match this specification will have their color changed to the current color.

## **colorscheme**

Apply a predefined color scheme to the group of atoms defined by the ASL spec.

Syntax:

**colorscheme** *scheme*=⟨text⟩ ⟨ASL⟩

Options:

*scheme*      The name of the current color scheme.

Valid values:    text strings

Default value:   **atype**

Operands:

⟨ASL⟩

A string in the atom specification language. All atoms which match this specification will have their color changed to match the currently selected color scheme.

## **combilibenum**

Start the job of Combinatorial Library Enumeration.

Syntax:

**combilibenum**

## **combilibenumaddattachment**

Adds an attachment to the core molecule using the given atoms.

Syntax:

**combilibenumaddattachment** *atom1*=⟨n⟩ *atom2*=⟨n⟩  
⟨attachment name⟩

Options:

*atom1*      The atom number of the atom in the original core to set as an attachment point. This is the atom which will be kept.

Valid values:    integers

Default value:    **1**

Minimum:        1

*atom2*      The atom number of the atom in the original core to set as an attachment point. This is the atom which will be removed.

Valid values:    integers

Default value:    **1**

Minimum:        1

Operands:

*<attachment name>*

The name of the attachment.

### **combilibenumclearreagentfile**

Clears the reagent file for the selected rows.

Syntax:

### **combilibenumclearreagentfile**

### **combilibenumdeleteattachment**

Deletes all of the selected attachments.

Syntax:

### **combilibenumdeleteattachment**

### **combilibenumexportdefinition**

Stores the current core molecule and attachments in a file.

Syntax:

**combilibenumexportdefinition** <file name>

Operands:

<file name>

The name of the file to store the core definition in.

**combilibenumimportdefinition**

Reads a core molecule and attachments from the given file.

Syntax:

**combilibenumimportdefinition** <file name>

Operands:

<file name>

The name of the file to read the core definition from.

**combilibenumoptions**

This command holds general options for Combinatorial Library Enumeration.

Syntax:

**combilibenumoptions** *untangle*=yes | no

Options:

- untangle* An option which allows post-combgen minimization (for library enumeration).  
Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **true**

## **combilibenumrefreshstructure**

This function refreshes the structure in the Workspace from the current core structure in CombiGlide.

Syntax:

**combilibenumrefreshstructure** *viewcappedcore=yes | no*

Options:

*viewcappedcore*

An option which allows viewing of the minimally capped core, rather than the original core, in the Workspace.

Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **false**

## **combilibenumrenameattachment**

Renames the attachment in CombiLibEnum to the new name.

Syntax:

**combilibenumrenameattachment** *row=<n> <new name>*

Options:

*row* The row to rename.

Valid values: integers

Default value: **1**

Minimum: **1**

Operands:

*<new name>*

The new name for the attachment.

## **combilibenumselectextendtablerow**

Extends the selection to this row in the attachments table in the CombiGlide Library Enumeration.

Syntax:

**combilibenumselectextendtablerow <row>**

Operands:

<row>

The row number to extend the select to.

**combilibenumselectonlytablerow**

Selects only this row in the attachments table in the CombiGlide Library Enumeration.

Syntax:

**combilibenumselectonlytablerow <row>**

Operands:

<row>

The row number to select only in the table row.

**combilibenumselecttablerow**

Selects the given row in the attachments table in the CombiGlide Library Enumeration.

Syntax:

**combilibenumselecttablerow <row>**

Operands:

<row>

The row number to select in the table.

## **combilibenumsetmolecule**

Sets the core molecule for the CombiLibEnum to the molecule containing the given atom.

Syntax:

**combilibenumsetmolecule** *title*=⟨text⟩ ⟨atom number⟩

Options:

*title* This option sets the title for the core molecule.

Valid values: text strings

Default value: **core**

Operands:

⟨atom number⟩

The atom number of the molecule.

## **combilibenumsetreagentfile**

Sets the reagent file for the selected rows.

Syntax:

**combilibenumsetreagentfile** ⟨reagent name⟩

Operands:

⟨reagent name⟩

The name of the reagent file.

## **combilibenumunselecttablerow**

Unselects the given row in the attachments table in the CombiGlide Library Enumeration.

Syntax:

**combilibenumunselecttablerow** <row>

Operands:

<row>

The row number to unselect in the table.

**compareatom**

Defines a single comparison atom to be used during a multiple minimization or conformational search. Comparison atoms are used to make comparisons in the process of determining if conformers are unique.

Syntax:

**compareatom** <atom>

Operands:

<atom>

The number of an atom which is to added to the list of comparison atoms during a conformational search.

**compareset**

Specifies a set of atoms to be used in comparisons of conformers during a multiple minimization or conformational search.

Syntax:

**compareset** <ASL>

Operands:

<ASL>

A string in the atom specification language. All atoms which match this description will be added to the list of comparison atoms.

## confelim

This keyword is used to set various options associated with starting Redundant Conformer Elimination jobs from Maestro.

Syntax:

```
confelim compare_in_place=yes | no energy_source=none | jaguar |
    mm2* | mm3* | amber* | opls* | amber94 | mmff | mmffs |
    oplsaa | opls2005 incorporate=append | replace | ignore
```

Options:

### *compare\_in\_place*

A boolean which controls whether ConfElim will compare structures without first doing a superposition.

Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **false**

### *energy\_source*

Source of energy for the comparisons

Valid values: none  
jaguar  
mm2\*  
mm3\*  
amber\*  
opls\*  
amber94  
mmff  
mmffs  
oplsaa  
opls2005

Default value: **opls2005**

### *incorporate*

How the results are to be incorporated into the project. This can be done with replacement of the existing entries or by appending as new entries to the project or by ignoring the final results.

Valid values: append  
replace  
ignore

Default value: **append**

## **confelimstart**

Start a Redundant Conformer Elimination job with the current settings.

Syntax:

### **confelimstart**

## **confelimwrite**

Write a Redundant Conformer Elimination input file with the current settings.

Syntax:

### **confelimwrite**

## **confgenltsearch**

Defines settings for ConfGen conformational searching in MacroModel.

Syntax:

```
confgenltsearch amidebonds=vary | retain | trans  
    compareatoms=none | heavy | heavy_polar_h  
    distinguishenantiomers=yes | no  limitsave=yes | no  
    maxdist=⟨x⟩  maxringconf=⟨n⟩  maxtorsdiff=⟨x⟩  
    numsave=⟨n⟩  numsteps=⟨n⟩  samplerings=yes | no  
    searchmode=standard | rapid | complete | thorough  
    usenumsteps=⟨n⟩  useseachmoves=yes | no  window=⟨x⟩
```

Options:

### *amidebonds*

This determines whether to vary amide bond conformation, retain original amide bond conformation, or set amide bond conformation to trans.

Valid values:    vary  
                  retain  
                  trans

Default value: **vary**

*compareatoms*

This determines which kinds of comparison atoms are automatically identified for judging redundant conformers.

Valid values:    none  
                  heavy  
                  heavy\_polar\_h  
Default value: **heavy\_polar\_h**

*distinguishenantiomers*

A boolean for whether to retain different enantiomers. If true, an additional opcode, NANT, is written to the .com file.

Valid values:    boolean (true|false; yes|no; y|n; on|off)  
Default value: **true**

*limitsave*    A boolean which controls whether number of structures saved is limited by numsave option (if true) or by numsteps (if false).

Valid values:    boolean (true|false; yes|no; y|n; on|off)  
Default value: **false**

*maxdist*    Maximum distance between atoms in equal structures.

Valid values:    reals  
Default value: **0.5**  
Minimum:        0.0

*maxringconf*

The maximum number of ring conformations generated by the search.

Valid values:    integers  
Default value: **8**  
Minimum:        1

*maxtorsdiff*

Maximum torsional angle difference between polar hydrogens in equal structures.

Valid values:    reals  
Default value: **60**  
Minimum:        0.0  
Maximum:        180.0

*numsave*    The number of structures that will be saved at the end of each search.

Valid values:    integers  
Default value: **1000**  
Minimum:        0

*numsteps*    An option which sets the number of steps which will be performed during the ConfGen conformational search.

Valid values: integers  
Default value: **1000**  
Minimum: 0

*samplerings*

A boolean which controls whether to sample rings.  
Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **true**

*searchmode*

This determines whether rapid (standard) or thorough (complete) search will be used.  
Valid values: standard  
rapid  
complete  
thorough  
Default value: **complete**

*usenumsteps*

An option which sets the number of steps which will be performed during the ConfGen conformational search.  
Valid values: integers  
Default value: **100**  
Minimum: 1

*usesearchmoves*

A boolean which controls whether number of search moves is limited by maxsearch moves option (if true).  
Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **true**

*window*

The energy window ( in kJ/mol ) within which structures will be saved.  
Valid values: reals  
Default value: **25**  
Minimum: 0.0

## **confgenmini**

Used to set values associated with a MacroModel energy minimization for Ligand Torsional Search (ConfGen)

Syntax:

```
configenmini converge=nothing | energy | gradient | movement  
method=sd | prcg | osvm | fmnr | tncg | lbfgs | optimal  
postmaxiter=<n> premaxiter=<n> threshold=<x>
```

Options:

*converge* This option determines which convergence criterion will be used during an energy minimization.

Valid values: nothing  
energy  
gradient  
movement

Default value: **gradient**

*method* This option determines which minimization method will be used.

Valid values: sd  
prcg  
osvm  
fmnr  
tncg  
lbfgs  
optimal

Default value: **tncg**

*postmaxiter*

This option determines the maximum number of iterations for post-minimization of generated structures.

Valid values: integers  
Default value: **50**  
Minimum: 0  
Maximum: 9999999

*premaxiter*

This option determines the maximum number of iterations for pre-minimization of input structures.

Valid values: integers  
Default value: **100**  
Minimum: 0  
Maximum: 9999999

*threshold* This option determines what the convergence threshold will be.

Valid values: reals  
Default value: **0.05**  
Minimum: 0.0

## configpotential

Set various options associated with the definition of the potential energy to be used in a MacroModel job.

Syntax:

```
configpotential cele=<x> charges=force_field | structure_file
    chnd=<x> cutoff=normal | extended | user_defined | none
    cvdw=<x> dielectric=<x> electrostatics=field_field | constant |
    distance_dependant field=mm2* | mm3* | amber* | opls* |
    amber94 | mmff | mmffs | oplsaa | opls2005 | opls2007
    solvent=none | water | chcl3 | octanol suppresshbond=yes | no
```

Options:

<i>cele</i>	This option determines what cutoff will be used for the electrostatic part of the energy calculation. Valid values:   reals Default value: <b>12</b> Minimum:       0.0 Maximum:      99999.0
<i>charges</i>	This option determines where the charges to be used in the energy calculation will come from. Valid values:   force_field structure_file Default value: <b>force_field</b>
<i>chnd</i>	This option determines what cutoff will be used for the hydrogen bond part of the energy calculation. Valid values:   reals Default value: <b>4</b> Minimum:       0.0 Maximum:      99999.0
<i>cutoff</i>	This option determines what type of non-bonded cutoff will be used in the energy calculation. Valid values:   normal extended user_defined none Default value: <b>normal</b>
<i>cvdw</i>	This option determines what VDW cutoff will be used in the energy calculation.

Valid values:    reals  
Default value: **7**  
Minimum:        0.0  
Maximum:        99999.0

*dielectric*    The dielectric constant to be used in the electrostatic part of the energy calculation.

Valid values:    reals  
Default value: **4**  
Minimum:        0.999999999

*electrostatics*

The electrostatic treatment to be used in the energy calculation.

Valid values:    field\_field  
                  constant  
                  distance\_dependant  
Default value: **distance\_dependant**

*field*          The force field to be used for the energy calculation.

Valid values:    mm2\*  
                  mm3\*  
                  amber\*  
                  opls\*  
                  amber94  
                  mmff  
                  mmffs  
                  oplsaa  
                  opls2005  
                  opls2007  
Default value: **opls2005**

*solvent*        The solvent model to be used for the energy calculation

Valid values:    none  
                  water  
                  chcl3  
                  octanol  
Default value: **none**

*suppresshbond*

A boolean which controls whether to suppress hydrogen bonding electrostatics.

Valid values:    boolean (true|false; yes|no; y|n; on|off)  
Default value: **true**

## **confgenreadpotential**

Read potential settings from a command file.

Syntax:

**confgenreadpotential** <file\_name>

Operands:

<file\_name>

The name of the file from which the potential settings will be read. If no name is specified, the default settings will be used.

## **confgenstart**

Start a Conformer Generation (Ligand Torsion Search) job with the current settings.

Syntax:

**confgenstart**

## **confgenwrite**

Write a Conformer Generation (Ligand Torsion Search) input file with the current settings.

Syntax:

**confgenwrite**

## **confsearch**

Defines settings for conformational searching in MacroModel.

Syntax:

```
confsearch distinguishenatiomers=yes | no enableauto=yes | no  
      max_distance_lmcs=⟨x⟩ maxdist=⟨x⟩ method=mcmm |  
      serial_mcmm | summ | mixed_lmcs | serial_mcmm_lmcs |  
      pure_lmcs | serial_lmcs | large_lmcs | mixed_large_lmcs  
      min_distance_lmcs=⟨x⟩ numsteps=⟨n⟩ numstructures=⟨n⟩  
      probability_tors_lmcs=⟨x⟩ torsionsampling=restricted |  
      intermediate | enhanced | extended usenumsteps=⟨n⟩  
      usesteps=yes | no window=⟨x⟩
```

Options:

*distinguishenatiomers*

This option determines to consider conformers more than once

Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **false**

*enableauto* Toggle whether or not to use AUTO setup for processing multiple structures.

Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **false**

*max\_distance\_lmcs*

The maximum possible distance for the fastest moving atom in each LMCS move.

Valid values: reals

Default value: **6**

Minimum: 0.0

*maxdist* Maximum distance between atoms in equal structures.

Valid values: reals

Default value: **0.25**

Minimum: 0.0

*method* This determines which method will be used to perform the conformational search.

Valid values: mcmm  
 serial\_mcmm  
 summ  
 mixed\_lmcs  
 serial\_mcmm\_lmcs  
 pure\_lmcs  
 serial\_lmcs  
 large\_lmcs  
 mixed\_large\_lmcs

Default value: **mcmm**

*min\_distance\_lmcs*

The minimum possible distance for the fastest moving atom in each LMCS move.

Valid values:   reals

Default value:   **3**

Minimum:       0.0

*numsteps* An option which sets the number of steps which will be performed during the conformational search.

Valid values:   integers

Default value:   **1000**

Minimum:       0

*numstructures*

The number of structures that will be saved at the end of each search.

Valid values:   integers

Default value:   **0**

Minimum:       0

*probability\_tors\_lmcs*

The probability that a TORS/MOLS move will be made during an LMCS search.

Valid values:   reals

Default value:   **0.5**

Minimum:       0.0

Maximum:       1.0

*torsionsampling*

Controls how sampling of torsions is done during the automatic setup for torsional sampling methods.

Valid values:   restricted  
                  intermediate  
                  enhanced  
                  extended

Default value:   **restricted**

*usenumsteps*

An option which sets the number of steps which will be performed during the conformational search.

Valid values:   integers

Default value:   **100**

Minimum:       1

*usesteps*

An option which sets the number of steps which will be performed during the conformational search.

Valid values:   boolean (true|false; yes|no; y|n; on|off)

Default value: **false**

*window*      The energy window ( in kJ/mol ) within which structures will be saved.

Valid values:    **reals**

Default value:    **50**

Minimum:        0.0

## **connect**

Connect atom pairs.

Syntax:

## **connect**

## **connectfuseatom**

Define an atom pair for which will be connected in a subsequent connect operation.

Syntax:

**connectfuseatom** <atom1> <atom2>

Operands:

<atom1> <atom2>

Two atom numbers which represent an atom pair to be connected by a subsequent connect or fuse command. The two atoms must be from different molecules and all connectfuseatom commands must specify molecules in the same order.

## **constrainedangle**

Specifies a angle between three atoms which is to be constrained by a harmonic constraint during a MacroModel calculation.

Syntax:

**constrainedangle** *angle*=⟨x⟩ *constant*=⟨x⟩ *width*=⟨x⟩ ⟨atom1⟩  
⟨atom2⟩ ⟨atom3⟩

Options:

*angle* The angle at which the atoms are to be constrained.

Valid values: reals

Default value: **-1**

*constant* The force constant for the harmonic constraint to be applied to the angle.

Valid values: reals

Default value: **100**

Minimum: 0.0

*width* The half-width for a flat-bottomed constraint.

Valid values: reals

Default value: **0**

Minimum: 0.0

Operands:

⟨atom1⟩ ⟨atom2⟩ ⟨atom3⟩

Three atoms which are to have the angle between them constrained by a harmonic constraint. The three atoms are not necessarily connected. Note that specifying a-b-c will be treated by the program as the same as specifying c-b-a.

## constrainedatom

Specifies a single atom to be constrained at its current positions during a MacroModel energy calculation. The atom will be constrained by the use of a harmonic constraint.

Syntax:

**constrainedatom** *constant*=⟨x⟩ *width*=⟨x⟩ ⟨atom\_number⟩

Options:

*constant* The force constant (in kJ/mol/angs) for the harmonic constraint.

Valid values:    reals  
Default value: **0**

*width*      The half-width of a flat bottomed restraint.

Valid values:    reals  
Default value: **0**  
Minimum:        0

Operands:

$\langle \text{atom\_number} \rangle$

The number of the atom to which a constraint is to be applied.

## constraineddist

Specifies a distance between two atoms which is to be constrained by a harmonic constraint during a MacroModel calculation.

Syntax:

**constraineddist** *constant*= $\langle x \rangle$  *distance*= $\langle x \rangle$  *remove\_nb*=yes | no  
*width*= $\langle x \rangle$   $\langle \text{atom1} \rangle$   $\langle \text{atom2} \rangle$

Options:

*constant*    The force constant for the harmonic constraint to be applied to the distance.  
Valid values:    reals  
Default value: **100**  
Minimum:        0.0

*distance*    The distance at which the atoms are to be constrained.

Valid values:    reals  
Default value: **-1**

*remove\_nb*    A boolean option which controls whether the non-bonded interaction (if any) between the two specified atoms will be removed.  
Valid values:    boolean (true|false; yes|no; y|n; on|off)  
Default value: **false**

*width*      The half-width for a flat-bottomed constraint.

Valid values:    reals  
Default value: **0**  
Minimum:        0.0

Operands:

$\langle \text{atom1} \rangle \langle \text{atom2} \rangle$

Two atoms which are to have the distance between them constrained by a harmonic constraint Note that specifying a-b will be treated by the program as the same as specifying b-a.

## **constrainedset**

Specifies a set of atoms to be constrained at their current positions. The atoms will be constrained by the use of a harmonic constraint.

Syntax:

**constrainedset** *constant*= $\langle x \rangle$  *width*= $\langle x \rangle$   $\langle \text{ASL} \rangle$

Options:

*constant*      The force constant (in kJ/mol/Angs) for the harmonic constraint.

Valid values:      reals

Default value:      **100**

Minimum:      0.0

*width*      The half-width of a flat bottomed restraint

Valid values:      reals

Default value:      **0**

Minimum:      0

Operands:

$\langle \text{ASL} \rangle$

The operand must be a valid string in the atom specification language. Any atoms which match will have constraints generated for them.

## **constrainedtorsion**

Specifies a torsion between four atoms which is to be constrained by a harmonic constraint during a MacroModel calculation.

Syntax:

**constrainedtorsion** *constant*=⟨x⟩ *torsion*=⟨x⟩ *width*=⟨x⟩  
⟨atom1⟩ ⟨atom2⟩ ⟨atom3⟩ ⟨atom4⟩

Options:

*constant* The force constant for the harmonic constraint to be applied to the torsion.

Valid values: reals

Default value: **100**

Minimum: 0.0

*torsion* The torsion at which the atoms are to be constrained.

Valid values: reals

Default value: **500**

*width* The half-width for a flat-bottomed constraint.

Valid values: reals

Default value: **0**

Minimum: 0.0

Operands:

⟨atom1⟩ ⟨atom2⟩ ⟨atom3⟩ ⟨atom4⟩

Four atoms which are to have the angle between them constrained by a harmonic constraint. Note that specifying a-b-c-d is treated by the program the same as specifying d-c-b-a.

## contactcriteria

Specify the criteria for calculating good, bad, ugly contacts. There is no upper-bound to C. C = {dist between the two atoms} divided by {vdW radius first atom + vdW radius of second atom}.

Syntax:

**contactcriteria** *bad*=⟨x⟩ *displaybad*=yes | no  
*displaygood*=yes | no *displayugly*=yes | no *good*=⟨x⟩  
*ugly*=⟨x⟩

Options:

<i>bad</i>	Criteria for bad contact
	Valid values:   reals
	Default value: <b>0.89</b>
	Minimum:       0.0
<i>displaybad</i>	If contact markers are shown, this option determines whether the bad contact markers will be displayed.
	Valid values:   boolean (true false; yes no; y n; on off)
	Default value: <b>true</b>
<i>displaygood</i>	If contact markers are shown, this option determines whether the good contact markers will be displayed.
	Valid values:   boolean (true false; yes no; y n; on off)
	Default value: <b>false</b>
<i>displayugly</i>	If contact markers are shown, this option determines whether the ugly contact markers will be displayed.
	Valid values:   boolean (true false; yes no; y n; on off)
	Default value: <b>true</b>
<i>good</i>	Criteria for good contact
	Valid values:   reals
	Default value: <b>1.3</b>
	Minimum:       0.0
<i>ugly</i>	Criteria for ugly contact
	Valid values:   reals
	Default value: <b>0.75</b>
	Minimum:       0.0

## contactset1

Specify the first set of atoms used in determining good, bad, and ugly contacts.

Syntax:

**contactset1** <ASL>

Operands:

<ASL>

A string in the atom specification language. Typical usage is to define contactset1 and contactset2. This set, contactset1, defines the “from” atoms. The contactset2 atoms define the “to” atoms. Contacts are calculated between these two sets. That is, the contacts are inter-set contacts. No intra-set contacts are calculated. If contactset2’s ASL string is empty, then contacts are calculated for all atoms in contactset1.

## **contactset2**

Specify the second set of atoms used in determining good, bad, and ugly contacts.

Syntax:

**contactset2** < ASL >

Operands:

< ASL >

A string in the atom specification language. Typical usage is to define contactset1 and contactset2. This set, contactset2, defines the “to” atoms. The contactset1 atoms define the “from” atoms. Contacts are calculated between these two sets. That is, the contacts are inter-set contacts. No intra-set contacts are calculated. If contactset2’s ASL string is empty, then contacts are calculated for all atoms in contactset1.

## **coupling**

Specifies a pair of atoms to have their coupling measured and displayed.

Syntax:

**coupling** < atom1 > < atom2 >

Operands:

< atom1 > < atom2 >

The two hydrogen atoms between which the coupling is to be measured. Note that specifying a-b is the same as specifying b-a

## **createlibrary**

Start the creation of the library in the Analyse Library step of CombiGlide.

Syntax:

## **createlibrary**

## **createpropsubset**

This command creates a property subset for the given table using the given property names.

Syntax:

**createpropsubset** <table> <propertynames>

Operands:

<table> <propertynames>

The number of the table to create the subset for. If the table operand is missing, no resize will be done. The names of the properties to use to create the subset.

## **createsubset**

Creates a subset in the project table using the currently selected entries. This function also switches to subset view.

Syntax:

## **createsubset**

## **defaultfc**

Requires a single operand which is the default force constant to be used for the constrainedatom command. The default value is used only when no value has been specified by the constant= option of the “constrainedatom” command.

Syntax:

**defaultfc** < default\_force\_constant\_value >

Operands:

< default\_force\_constant\_value >

The value of the default force constant in kJ/mol/Angstrom

## delete

Delete a named object. The object type is the same as the command which is used to create that type of object. For example to delete a set named “set1” use: delete set set1. All instances of any type of object can be deleted with the “all” name. There are two special objects: “atom” and “bond” for deleting atoms and bonds respectively

For example: delete set set1

delete constrainedatom 1

delete set all delete res. 1-5 delete bond 1 2

Syntax:

**delete** *includeterminal*=yes | no < object\_type >  
< object\_name >|all|< ASL >|< bond >

Options:

*includeterminal*

This option is used to control the behavior when deleting atoms and bonds. If this is “true” then all terminally attached atoms will also be deleted when deleting bonds and atoms.

Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **true**

Operands:

< object\_type > < object\_name >|all|< ASL >|< bond >

The first operand determines what type of object is to be deleted. This can be the name of any of the named objects in Maestro (sets, filters, holds etc.) or can be “atom” or “bond”. The subject operands depend on the first operand. If the first operand names an object then the second operand must be the name which was specified when that object was created. If the first operand is “atom” then the second operand must be a valid string in

the atom specification language and all atoms which match that string will be deleted. Finally if the first operand is “bond” then the second and third operands are expected to be the numbers of the two atoms which define that bond.

## **deleteproperty**

This is a standard alias for **propertydelete** (see [propertydelete], page 369).

## **dialogoptions**

Sets options for dialog boxes.

Syntax:

**dialogoptions** *show=yes | no*

Options:

*show* If this option is set to false, then dialog boxes will not be displayed.

Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **true**

## **dihedral**

Specifies a quartet of atoms to have their dihedral angle measured and displayed.

Syntax:

**dihedral** <atom1> <atom2> <atom3> <atom4>

Operands:

<atom1> <atom2> <atom3> <atom4>

The four atoms between which the dihedral angle is to be measured. Note that specifying a-b-c-d is the same as specifying d-c-b-a

## dihedraldrive

Used to set a dihedral angle to be driven. The four operands are the atom numbers defining the dihedral to be driven. A maximum of two angles can be driven.

Syntax:

```
dihedraldrive finish=⟨x⟩ increment=⟨x⟩ start=⟨x⟩ ⟨atom1⟩  
⟨atom2⟩ ⟨atom3⟩ ⟨atom4⟩
```

Options:

*finish*      Specifies the finishing angle for the dihedral drive.

Valid values:    reals

Default value: **360**

*increment*    Specifies the angle increment for the dihedral drive.

Valid values:    reals

Default value: **30**

*start*        Specifies the starting angle for the dihedral drive.

Valid values:    reals

Default value: **0**

Operands:

```
⟨atom1⟩ ⟨atom2⟩ ⟨atom3⟩ ⟨atom4⟩
```

The four operands are treated as atom numbers which define the dihedral to be driven.

## displayatom

Display all atoms in the set described by the ASL operand.

Syntax:

```
displayatom ⟨ASL⟩
```

Operands:

```
⟨ASL⟩
```

A string in the atom specification language which describes the set of atoms which are to be displayed.

## **displayonlyatom**

Display only the set described by the ASL operand.

Syntax:

**displayonlyatom** ⟨ ASL ⟩

Operands:

⟨ ASL ⟩

A string in the atom specification language which describes the set of atoms which are to be displayed.

## **displayonlypose**

This command undisplays all poses in the pose viewer, then displays the specified structure from the pose set.

Syntax:

**displayonlypose** ⟨ pose\_name ⟩

Operands:

⟨ pose\_name ⟩

The name of the pose structure.

## **displayopt**

Set properties for the display, such as the background color.

Syntax:

```
displayopt adjustclip=yes | no aligndepth=<x> bgcindex=<n>
bgcolor=<text> eye_sepf=<x> fog=auto | on | true | yes | off
| false | no fogatomlabels=yes | no fogcutoff=<n>
fogdensity=<x> fogendz=<x> fogstartz=<x> fogtype=linear |
exp | exp2 maximize=yes | no perspective=yes | no
perspectivescale=<x> sepf=<x> sizef=<x> start_str=<text>
stereo=yes | no stereomethod=hardware | crosseyed | walleyed
| fullscreen stop_str=<text>)
```

Options:

*adjustclip* A boolean option which determines whether to adjust clipping to preserve stereo depth.

Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **true**

*aligndepth* Relative z-depth (1 for back clipping plane, 0 for front clipping plane) at which stereo images appear to converge. Moves region inside clipping planes in or out of the screen.

Valid values: reals

Default value: **0.5**

Minimum: 0.0

Maximum: 1.0

*bgcindex* An integer which indicates color index which is to be used for the background

Valid values: integers

Default value: **1**

Minimum: 1

Maximum: 256

*bgcolor* A string which is the color name for background color. Valid color names are described in the file \$SCHRODINGER/maestro-vX.X/data/res/colors.res

Valid values: text strings

Default value:

*eye\_sepf* Eye separation for stereo as fraction of normal. Actual separation may be made smaller to limit maximum stereo disparity to about 1 inch at clipping planes.

Valid values: reals

Default value: **1**

Minimum: 0.0

Maximum: 2.0

*fog* Set fog to be displayed, to not be displayed, or to toggle automatically based on the number of atoms in the workspace.

Valid values: auto  
 on  
 true  
 yes  
 off  
 false  
 no

Default value: **auto**

*fogatomlabels*

A Boolean option which determines whether to display fog on atom labels (only displayed if fog is already on in general).

Valid values: boolean (true|false; yes|no; y|n; on|off)  
 Default value: **true**

*fogcutoff* Specifies the minimum number of atoms in the workspace for which fog will be displayed, when fogging is set to auto.

Valid values: integers  
 Default value: **40**  
 Minimum: 0

*fogdensity* Factor controlling optical density of fog.

Valid values: reals  
 Default value: **1.5**  
 Minimum: 0.0  
 Maximum: 4.0

*fogendz* Factor controlling end of linear fog ramp.

Valid values: reals  
 Default value: **1**  
 Minimum: 1.0  
 Maximum: 2.0

*fogstartz* Factor controlling start of linear fog ramp.

Valid values: reals  
 Default value: **0.35**  
 Minimum: 0.0  
 Maximum: 0.999

*fogtype* Specifies method used for calculating fog.

Valid values: linear  
 exp  
 exp2  
 Default value: **linear**

*maximize* A boolean option which determines whether to maximize the display window (on) or restore to the original window size (off)

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **false**

*perspective*

A boolean option which determines whether to do perspective, rather than orthogonal, projection.

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **false**

*perspectivescale*

Ratio between the actual size and the projected size of an object placed 1 foot behind display screen. This has a corresponding viewing distance,  $\text{view\_dist} = (1 \text{ ft}) / (\text{perspectivescale} - 1)$ . For a known viewing distance,  $\text{perspectivescale} = 1 + (1 \text{ ft} / \text{view\_dist})$ . Therefore, for a normal viewing distance of 2 feet,  $\text{perspectivescale} = 1 + (1 \text{ ft} / 2 \text{ ft}) = 1.5$ .

Valid values: reals  
Default value: **1.5**  
Minimum: 1.0  
Maximum: 4.0

*sepf*

Image separation as fraction of maximum horizontal separation (window width minus image width)

Valid values: reals  
Default value: **0**  
Minimum: 0.0  
Maximum: 1.0

*sizef*

Image size as fraction of available width

Valid values: reals  
Default value: **1**  
Minimum: 0.0  
Maximum: 1.0

*start\_str*

The system command to start hardware stereo.

Valid values: text strings  
Default value:

*stereo*

A boolean option which determines whether to display in stereo

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **false**

*stereomethod*

Specifies method used for stereo viewing

Valid values: hardware  
crosseyed  
walleyed  
fullscreen

Default value: **hardware**

*stop\_str* The system command to stop hardware stereo.

Valid values: text strings

Default value:

## **displaypose**

This command retrieves the pose set structure in the pose viewer and adds it to the current on-screen structure, after removing atoms belonging to the pose structure from the workspace.

Syntax:

**displaypose** <pose\_name>

Operands:

<pose\_name>

The name of the pose structure.

## **distance**

Specifies a pair of atoms to have their distance measured and displayed.

Syntax:

**distance** <atom1> <atom2>

Operands:

<atom1> <atom2>

The two atoms between which the distance is to be measured. Note that specifying a-b is the same as specifying b-a

## **distancecheck**

Specifies two atoms which define a distance to be checked during a conformational search.

Syntax:

**distancecheck** *allowed*=⟨x⟩ *value*=⟨x⟩ ⟨atom1⟩ ⟨atom2⟩

Options:

*allowed* The maximum allowed variation from the “target” value during the distance check.

Valid values: reals

Default value: **0**

Minimum: 0.0

*value* The target value for the distance check (Angstroms)

Valid values: reals

Default value: **0**

Minimum: 0.0

Operands:

⟨atom1⟩ ⟨atom2⟩

The numbers of two atoms which are to be considered are to have their distance checked during a conformational search. Note that specifying a-b is the same as specifying b-a.

## dynamics

Used to set options associated with MacroModel molecular or stochastic dynamics procedures.

Syntax:

**dynamics** *equilibration*=⟨x⟩ *method*=molecular | stochastic  
  *sample*=⟨n⟩ *shake*=nothing | hydrogens | all *simulation*=⟨x⟩  
  *temperature*=⟨x⟩ *timestep*=⟨x⟩

Options:

*equilibration*

The time used for equilibration for the equilibration part of the simulation in ps.

Valid values: reals

Default value: **1**

Minimum: 0.0

<i>method</i>	This option determines which dynamics method will be used. Valid values: molecular stochastic Default value: <b>stochastic</b>
<i>sample</i>	The number of structures to be sampled during the simulation. Valid values: integers Default value: <b>0</b> Minimum: 0
<i>shake</i>	This option determines how the SHAKE algorithm will be applied during the simulation. Valid values: nothing hydrogens all Default value: <b>nothing</b>
<i>simulation</i>	The total time to be used in the simulation. Valid values: reals Default value: <b>10</b> Minimum: 0.0
<i>temperature</i>	The temperature at which the simulation is to be run in Kelvin. Valid values: reals Default value: <b>300</b> Minimum: 0.0
<i>timestep</i>	The timesetp which is to be used for the simulation in fs. Valid values: reals Default value: <b>1.5</b> Minimum: 0.0

## ecalc

This is a standard alias for **calcenergy** (see [calcenergy], page 47).

## ejob

This keyword is used to set various options associated with running Macro-Model jobs.

Syntax:

```
ejob host=<text> incorporate=append | replace | ignore  
      input_file=<text> job=<text> login=<text>  
      rce_structure_source=selected_entries | workspace | file  
      structure_source=selected_entries | workspace | file
```

Options:

*host*      The name of the host for the MacroModel job.

Valid values:    text strings

Default value:

*incorporate*

How the results are to be incorporated into the project. This can be done with replacement of the existing entries or by appending as new entries to the project or by ignoring the final results.

Valid values:    append  
                  replace  
                  ignore

Default value: **append**

*input\_file*    The name of the structure input file.

Valid values:    text strings

Default value:

*job*        The name for the MacroModel job.

Valid values:    text strings

Default value: **mmodtmp**

*login*       The login name under which a MacroModel will be run.

Valid values:    text strings

Default value:

*rce\_structure\_source*

Whether to use the selected entries in the current project, or a specified file with multiple structures as structure input for the job.

Valid values:    selected\_entries  
                  workspace  
                  file

Default value: **selected\_entries**

*structure\_source*

Whether to use the selected entries in the current project or what is in the workspace as input for the job.

Valid values: selected\_entries  
workspace  
file  
Default value: **workspace**

## elementlabels

Set the element label options.

Syntax:

**elementlabels** *labelscheme*=heterohydro | hetero | off

Options:

*labelscheme*

Three Options for displaying element labels: 1) on all non-carbon atoms, including hydrogen. 2) on all non-carbon atoms, except hydrogen (i.e., not on hydrogen) 3) off (no element labels displayed).

Valid values: heterohydro  
hetero  
off

Default value: **off**

## embrace

Used to control the eBMrAcE feature of MacroModel.

Syntax:

**embrace** *energymode*=interaction | energydiff *inputfile*=⟨ text ⟩  
*ligandsmode*=inputfile | selectedentries *output*=complexes |  
ligands | all *receptor*=⟨ text ⟩ *receptormode*=firstitem | entry

Options:

*energymode*

This option determines which energy mode will be used by eBMrAcE.

Valid values: interaction  
energydiff  
Default value: **energydiff**

*inputfile* The name of the eMBrAcE ligands input file.

Valid values: text strings  
Default value:

*ligandsmode*

This option determines which source of ligands will be used by eBMrAcE.

Valid values: **inputfile**  
selectedentries  
Default value: **inputfile**

*output* This option determines what structural output will be produced.

Valid values: complexes  
ligands  
all  
Default value: **complexes**

*receptor* This option specifies which entry in the project is to be treated as the receptor.

Valid values: text strings  
Default value:

*receptormode*

This option determines which kind of receptor will be used by eBMrAcE.

Valid values: firstitem  
entry  
Default value: **firstitem**

## **embracecsearch**

Defines settings for eMBrAcE conformational searching in MacroModel.

Syntax:

**embracecsearch** *max\_distance\_lmcs=⟨x⟩ maxdist=⟨x⟩ method=embrace\_mcmm | embrace\_lmcs | embrace\_mixed\_lmcs min\_distance\_lmcs=⟨x⟩ numconfs=⟨n⟩ numsteps=⟨n⟩ numstructures=⟨n⟩ probability\_tors\_lmcs=⟨x⟩ window=⟨x⟩*

Options:

*max\_distance\_lmcs*

The maximum possible distance for the fastest moving atom in each LMCS move.

Valid values: reals

Default value: **6**

Minimum: 0.0

*maxdist* Maximum distance between atoms in equal structures.

Valid values: reals

Default value: **0.25**

Minimum: 0.0

*method* This determines which method will be used to perform the eM-BrAcE conformational search.

Valid values: embrace\_mcmm

embrace\_lmcs

embrace\_mixed\_lmcs

Default value: **embrace\_mcmm**

*min\_distance\_lmcs*

The minimum possible distance for the fastest moving atom in each LMCS move.

Valid values: reals

Default value: **3**

Minimum: 0.0

*numconfs* The number of input conformations available to seed each search.

Valid values: integers

Default value: **1**

Minimum: 0

*numsteps* An option which sets the number of steps which will be performed during the eMBrAcE conformational search.

Valid values: integers

Default value: **100**

Minimum: 0

*numstructures*

The number of structures that will be saved at the end of each search.

Valid values: integers  
Default value: **1**  
Minimum: 0

*probability.tors\_lmcs*

The probability that a TORS/MOLS move will be made during an LMCS search.

Valid values: reals  
Default value: **0.5**  
Minimum: 0.0  
Maximum: 1.0

*window* The energy window ( in kJ/mol ) within which structures will be saved.

Valid values: reals  
Default value: **500**  
Minimum: 0.0

## **endundoblock**

End the current undoable command block.

Syntax:

## **endundoblock**

## **energykill**

Specifies a running job to be killed. This will force the program running the job to be stopped.

Syntax:

**energykill** <job\_id> | <job\_name> | <job\_name> <project\_name>

Operands:

<job\_id> | <job\_name> | <job\_name> <project\_name>

The ID or name of the job which is to be put to killed. This must be the ID or name of a job which is currently running or sleeping.

Aliases:

**kill** (see [kill], page 217)

## **energymonitor**

Specifies a job to monitor.

Syntax:

**energymonitor** <job\_id> | <job\_name> | <job\_name>  
                          <project\_name>

Operands:

<job\_id> | <job\_name> | <job\_name> <project\_name>

The ID or name of the job which is to be monitored. If no operand is specified then the job which was most recently monitored will be monitored.

Aliases:

**monitor** (see [monitor], page 242)

## **energysleep**

Specifies a job to be put to sleep

Syntax:

**energysleep** <job\_id> | <job\_name> | <job\_name> <project\_name>

Operands:

<job\_id> | <job\_name> | <job\_name> <project\_name>

The ID or name of the job which is to be put to sleep. This must be the ID or name of a job which is currently running.

Aliases:

**sleep** (see [sleep], page 440)

## **energystart**

Start a MacroModel job with the current energy settings.

Syntax:

**energystart** *onscreen=yes | no <job\_name>*

Options:

*onscreen*      Determines whether the on-screen structure will be written and used as the .dat file for the MacroModel job.

Valid values:      boolean (true|false; yes|no; y|n; on|off)

Default value:      **false**

Operands:

*<job\_name>*

The name of the job which is to be started. This operand is not the name of a file and should not contain any suffix. From this name the files “*job\_name*”.com and “*job\_name*”.mae will be created.

## **energystop**

Specifies a running job to be stopped. The program running the job will be requested to stop but may take some time to respond to the command.

Syntax:

**energystop** *<job\_id> | <job\_name> | <job\_name> <project\_name>*

Operands:

*<job\_id> | <job\_name> | <job\_name> <project\_name>*

The ID or name of the job which is to be stopped. This must be the ID or name of a job which is currently running or sleeping.

Aliases:

**stop** (see [stop], page 441)

## **energytask**

Determines which energy task is currently being set up.

Syntax:

**energytask** ecalc|mini|moldyn|drive|mult|csearch|mcsd|minta|  
embrace|embracecsearch

Operands:

ecalc|mini|moldyn|drive|mult|csearch|mcsd|minta|  
embrace|embracecsearch

The type of energy task which is to be set up. The operand cannot be abbreviated and must be given in full.

## **energyupdate**

Specifies a running job to be updated.

Syntax:

**energyupdate** ⟨job\_id⟩ | ⟨job\_name⟩ | ⟨job\_name⟩  
⟨project\_name⟩

Operands:

⟨job\_id⟩ | ⟨job\_name⟩ | ⟨job\_name⟩ ⟨project\_name⟩

The ID or name of the job which is to be put to updated. This is not the name of a file and no suffix is required. At present only MacroModel conformational searches will pay any attention to a request to update, however it is no error for an update to be issued for any type of job.

Aliases:

**update** (see [[update](#)], page 471)

## **energyupdatejobstatus**

update the job status of all the jobs in the monitor panel.

Syntax:

## **energyupdatejobstatus**

### **energywake**

Specifies a sleeping job to be woken up.

Syntax:

**energywake** <job\_id> | <job\_name> | <job\_name> <project\_name>

Operands:

<job\_id> | <job\_name> | <job\_name> <project\_name>

The ID of the job which is to be put to woken up. This must be the ID of a job which is currently sleeping.

Aliases:

**wake** (see [[wake](#)], page 500)

## **enhance3d**

Fit clipping planes to displayed structure. If display stereo is on, this may help prevent conflicts between the clipping range and the stereo angle that lead to clipping of the displayed structure or flatter appearance of the structure in stereo, especially when zooming in. If display perspective is on, this may help prevent conflicts between the front clipping plane and the viewing distance (which corresponds to the perspective scale) that lead to clipping of the displayed structure or reduced perspective scaling, especially when zooming in. If display fog is on, this helps maximize the shading contrast between near and far atoms and helps make the front part of the displayed structure more visible.

Syntax:

## **enhance3d**

## **entrycombine**

Combine selected project entries into a single entry.

Syntax:

**entrycombine** *replace=yes | no <new\_name> <ESL>*

Options:

*replace* This determines whether the combined entry is allowed to replace an existing entry having the name <new\_name>. If the name of the combined entry matches the name of an entry in the workspace, the entry in the workspace is also replaced. If the replace option is off, the user will be asked to accept a unique name based upon <new\_name>.

Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **false**

Operands:

<new\_name> <ESL>

The name to be given to the single combined project entry, followed by a valid ESL expression. If no name is specified, no rename will be done. The ESL expression specifies which entries are to be combined.

## **entrycopyprop**

Set property values for selected project entries from another property. The to property value will be copied from the from property for the entries selected by the ESL expression. If the to property does not already exist, it will be created. When the data type for the two properties is not the same, a reasonable attempt is made to convert between them. If a selected entry has no value for the from property, or the conversion from a string value fails, the value is generally cleared for the to value for that entry. Entry names are never cleared.

Syntax:

**entrycopyprop** *from=<text> to=<text> <ESL>*

Options:

*from* The name of the property which provides the values to be copied. This can be either the user name or the m2io data name for the property.

Valid values: text strings

Default value:

**to** The name of the property to be modified. This can be either the user name or the m2io data name for the property, if the to property exists. If the to property does not exist, it will be created using the data type of the from property, with user as the author. If the property is the entry name (e.g. Entry Name or s\_m\_entry\_name ), an entryrename with replace=no will be done. If the property is the 'included in Workspace' property (In or b\_m\_entry\_in\_workspace ), then entrywsinclude or entrywsexclude will be done, if needed.

Valid values: text strings  
Default value:

Operands:

**<ESL>**

The ESL expression specifies for which entries the property values are to be copied.

## **entrydelete**

Delete each selected project entry.

Syntax:

**entrydelete <ESL>**

Operands:

**<ESL>**

A valid ESL specification. Delete those entries which match the ESL description.

## **entrydisassociate**

Create a new entry for each molecule in each selected project entry. Each of these new entries will be given a unique name, based upon the name of its originating entry. Any of the originating entries which are in the workspace are replaced by their disassociated molecule entries.

Syntax:

**entrydisassociate** <ESL>

Operands:

<ESL>

A valid ESL specification. Disassociate those entries which match the ESL description.

## **entrydragselection**

Move current entry selection to the specified table row.

Syntax:

**entrydragselection** <row> <table>

Operands:

<row> <table>

The destination row number. In place of the row number, “top” may be used to specify the first row of the table (row 1) and “bottom” may be used to specify placement after the last unselected row (last\_row + 1). The selected entries will be moved as a block, placing the first selected entry below the first unselected entry which is above the destination row. If there is no unselected entry above the destination row, the selected entries will be moved to the top of the table. The name of the table to use as the source for the row number. If the table operand is missing, the current or default table (1) will be used.

## **entryduplicate**

Create a new duplicate entry for each selected project entry. Each of these new entries will be given a unique name, based upon the name of its originating entry.

Syntax:

## **entryduplicate <ESL>**

Operands:

<ESL>

A valid ESL specification. Duplicate those entries which match the ESL description.

## **entryexport**

Export selected entries from the current project to a file or a number of files. If the export command is issued with options, but without a filename then nothing is written but the options are updated. An export command with both options and a file name specified will result in the export being performed with the new options to the specified file(s). When exporting each entry to an individual file, the given file name will be the base file name which is used to name those individual files by adding their associated postfixes to the base file name.

Syntax:

```
entryexport append=yes | no displayedonly=yes | no files=single
    | individual format=mmod | pdb | mol2 | maestro | sd |
    | jaguarinput | gaussinput | biograf | xyz | jaguaroutput |
    | gaussian92 | gaussian94 | gamess | mopaccartesian |
    | mopacinternal | mopacoutput | babelpdb | mdl | babelmol |
    | babelmol2 | spartan | spartansemi | spartanmm | gamessinput |
    | gausscartesian | gaussian92 | jaguarzmatrix | jaguarcartesian |
    | any | reagentprep graphical=yes | no names=withentry |
    | withnumber | entry source=workspace | selected [<filename>]
```

Options:

*append*      This option determines whether to append to the file which is going to be written. It will be grayed out when PDB format is selected.

Valid values:    boolean (true|false; yes|no; y|n; on|off)  
Default value:    **false**

*displayedonly*

If this option is set to true, then only the displayed atoms in the Workspace will be exported.

Valid values:    boolean (true|false; yes|no; y|n; on|off)  
Default value:    **false**

<i>files</i>	This option sets the number of export files. The two options are: export entries to a single file (0 default) or export each entry to an individual file.  Valid values: <b>single</b> <b>individual</b> Default value: <b>single</b>
<i>format</i>	This option sets the format of the file to be written. Valid values are “mmod”, “pdb”, “mol2”, “maestro” or “sd”.  Valid values:    mmod pdb mol2 maestro sd jaguarinput gaussinput biograf xyz jaguaroutput gaussian92 gaussian94 gamess mopaccartesian mopacinternal mopacoutput babelpdb mdl babelmol babelmol2 spartan spartansemi spartanmm gamessinput gausscartesian gaussianz jaguarzmatrix jaguarcartesian any reagentprep Default value: <b>maestro</b>
<i>graphical</i>	This option determines whether to export graphical information when an entry is being saved to disk.  Valid values:    boolean (true false; yes no; y n; on off) Default value: <b>true</b>

*names* This option sets output file names when a number of files are exported. There are three options: file name + entry name, file name + number, just entry name. This option will be grayed out if only one file is exported.

Valid values: withentry  
withnumber  
entry  
Default value: **withentry**

*source* Whether to export selected entries in the current project table or to export what is in the workspace, including scratch entry.

Valid values: workspace  
selected  
Default value: **selected**

Operands:

[⟨ filename ⟩]

The name of the file to which entries will be written. If no name is specified, then no export will be done.

## **entryexportspreadsheet**

Export selected entries from the current project to a file for use in a spreadsheet. This can be comma-separated value (.csv) format or tab-delimited format.

Syntax:

```
entryexportspreadsheet columns=all | subset delimiter=<text>  
rows=all | subset | selected | included <filename>
```

Options:

*columns* This option sets the columns to be output. The columns can either be all or subset .

Valid values: all  
subset  
Default value: **all**

*delimiter* This option sets the delimiter to use to separate columns.

Valid values: text strings  
Default value: ,

*rows*      This option sets the rows to be written out. The rows can be all , subset , selected , or included .

Valid values:    all  
                  subset  
                  selected  
                  included

Default value: **all**

Operands:

*filename*

The name of the file to which entries will be written. If no name is specified, then no export will be done.

## **entryextendselection**

Extend current entry selection to encompass the specified entry.

Syntax:

**entryextendselection** <entry\_name>

Operands:

<entry\_name>

The name of an entry in the current project. A range selection will be done between currently selected entries and the specified entry.

## **entryextendwsinclude**

Extends the included entries to encompass the selected project entry.

Syntax:

**entryextendwsinclude** <ESL>

Operands:

<ESL>

A valid ESL specification. Extends included entries to encompass the entries which match the ESL description into the workspace.

## **entrygroupcollapse**

Collapses the given group.

Syntax:

**entrygroupcollapse** <group\_name>

Operands:

<group\_name>

Group name. Collapses the given group and hides all the entries in that group.

## **entrygroupcreate**

Creates a group for given entries.

Syntax:

**entrygroupcreate** <group\_name> <ESL>

Operands:

<group\_name> <ESL>

The new name to be given to the group of entries that match given ESL expression. The name should be unique. The new group name should be followed by a valid ESL expression to specify which entries are to be grouped. If no entry matches the ESL expression, no new group will be created. No empty group can exists.

## **entrygroupdelete**

Deletes the given group and all the entries in it.

Syntax:

**entrygroupdelete** <group\_name>

Operands:

<group\_name>

Name of the group to be deleted. Deletes the given group and all the entries in that group.

### **entrygroupduplicate**

Creates a new duplicate entry for each entry of the source group. Each of these new entries will be given a unique name, based upon the name of its originating entry. All these newly created entries will be grouped under a new group; and that new group will be given a unique name, based upon the name of source group.

Syntax:

**entrygroupduplicate** <group\_name>

Operands:

<group\_name>

The name of the group to be duplicated. Duplicates the given group along with all its entries.

### **entrygroupexpand**

Expands the given group.

Syntax:

**entrygroupexpand** <group\_name>

Operands:

<group\_name>

Group name. Expands the given group and displays all the entries in that group in PT.

### **entrygroupexpandonly**

Expands the given group and collapses all other groups

Syntax:

**entrygroupexpandonly** <group\_name>

Operands:

<group\_name>

Group name. Expands the given group, display all the entries in that group in PT and collapses all the other groups.

## **entrygroupextendselection**

Extend current entry selection to encompass all the entries in the specified group.

Syntax:

**entrygroupextendselection** <group\_name>

Operands:

<group\_name>

The name of a entry group in the current project. A range selection will be done between currently selected entries and the specified group; all the entries in that group will also be selected.

## **entrygroupmove**

Moves the given entry group to the specified table row.

Syntax:

**entrygroupmove** <group\_name> <row>

Operands:

<group\_name> <row>

Name of the group to be moved. Group name and the destination row number. In place of the row number, “top” may be used to specify the first row of the table (row 1) and “bottom” may be used to specify placement after the last row (last\_row + 1). If the target row corresponds to an ungrouped entry, the group will be placed just after that last ungrouped entry section.

If the target row corresponds to a grouped entry, the group will be placed just before the group to which the target entry belongs to. All the entries of the group will be moved as a block.

## **entrygrouprename**

Rename the given group.

Syntax:

**entrygrouprename** <old\_name> <new\_name>

Operands:

<old\_name> <new\_name>

The name of the group which we want to rename followed by the new name for the group.

## **entrygroupselect**

Select all the entries of given group.

Syntax:

**entrygroupselect** <group\_name>

Operands:

<group\_name>

Group name. Selects all the entries of given group does not change the selection state of the other entries.

## **entrygroupselectonly**

Selects the all entries of the given group and unselects all other entries.

Syntax:

**entrygroupselectonly** <group\_name>

Operands:

<group\_name>

Group name. Selects all the entries of given group and unselects all other entries.

**entrygroupsettitle**

Set the title for the given group.

Syntax:

**entrygroupsettitle** <group\_name> <title>

Operands:

<group\_name> <title>

The name of the group whose title we want to set followed by the title.

**entrygroupungroup**

Ungroups the entries in the given group.

Syntax:

**entrygroupungroup** <group\_name>

Operands:

<group\_name>

<group\_name> Remove the group and move all the entries in that group to the end of ungrouped section. In project table, all the ungrouped entries (if any) will be present at the top i.e. before all the groups.

**entrygroupunselect**

Unselects all the entries of given group.

Syntax:

**entrygroupunselect** <group\_name>

Operands:

<group\_name>

Group name. Unselectis all the entries of given group and does not change the selection state of the other entries.

**entrygroupwsexclude**

Exclude all the entries of given group from the workspace.

Syntax:

**entrygroupwsexclude** <group\_name>

Operands:

<group\_name>

A valid group name. Excludes those entries which belong to the given group from the workspace.

**entrygroupwsinclude**

Include all entries of given group into the workspace.

Syntax:

**entrygroupwsinclude** <group\_name>

Operands:

<group\_name>

A valid group name. Include all the entries which belong to the given group.

## **entrygroupwsincludeonly**

Include only the entries of given group into the workspace. Exclude all others.

Syntax:

**entrygroupwsincludeonly** <group\_name>

Operands:

<group\_name>

A valid ESL group name. Include only those entries which belong to the given group and exclude all other entries.

## **entryimport**

Import structures into the current project.

Syntax:

**entryimport** *all=yes | no* *allfiles=yes | no* *end=yes | no*  
*format=mmod | pdb | mol2 | maestro | sd | jaguarinput |*  
*gaussinput | biograf | xyz | jaguaroutput | gaussian92 |*  
*gaussian94 | gamess | mopaccartesian | mopacinternal |*  
*mopacoutput | babelpdb | mdl | babelmol | babelmol2 |*  
*spartan | spartansemi | spartanmm | gamessinput |*  
*gausscartesian | gaussian92 | jaguarzmatrix | jaguarcartesian |*  
*any | reagentprep* *graphical=yes | no* *start=<n>* *total=<n>*  
*wsinclude=none | first | all* *wsreplace=yes | no* [*(filename)*]

Options:

- |                 |   |
|-----------------|---|
| <i>all</i>      | This determines whether all structures will be imported, or just a specified range.<br>Valid values: boolean (true false; yes no; y n; on off)<br>Default value: <b>true</b>        |
| <i>allfiles</i> | This determines whether all files in the directory given in the operand will be imported.<br>Valid values: boolean (true false; yes no; y n; on off)<br>Default value: <b>false</b> |

<i>end</i>	This determines if all structures in the file are to be imported starting from the structure specified by start. Valid values: boolean (true false; yes no; y n; on off) Default value: <b>false</b>
<i>format</i>	This option sets the format of the file to be read. Valid values are “maestro”, “mmod”, “pdb”, “mol2”, or “sd”. Valid values: mmod pdb mol2 maestro sd jaguarinput gaussinput biograf xyz jaguaroutput gaussian92 gaussian94 gamess mopaccartesian mopacinternal mopacoutput babelpdb mdl babelmol babelmol2 spartan spartansemi spartanmm gamessinput gausscartesian gaussianz jaguarzmatrix jaguarcartesian any reagentprep Default value: <b>maestro</b>
<i>graphical</i>	This determines whether the file will be read with just the geometric information or if the visual information in the file will be read and used for displaying Valid values: boolean (true false; yes no; y n; on off) Default value: <b>true</b>

<i>start</i>	This option sets the number of the first structure to be imported, if not importing all.  Valid values: integers Default value: <b>1</b> Minimum: 1
<i>total</i>	The total number of structures to be imported from the file, if not importing all structures.  Valid values: integers Default value: <b>1</b> Minimum: 1
<i>wsinclude</i>	This option determines which of the imported structures are to be included in the workspace. Valid values are “none”, “first”, or “all”.  Valid values: none first all Default value: <b>first</b>
<i>wsreplace</i>	This determines whether the structures currently in the workspace will be replaced by the included imported structures, or whether they will be included in the workspace also.  Valid values: boolean (true false; yes no; y n; on off) Default value: <b>true</b>

Operands:

[⟨ filename ⟩]

The name of the file from which structures will be imported. If no name is specified, then no import will be done.

## **entryimportspreadsheet**

Import entries from a file into the project table.

Syntax:

**entryimportspreadsheet** *delimiter*=comma | tab | userdefined  
  *userdelimiter*=⟨ text ⟩ ⟨ filename ⟩ ⟨ import\_key ⟩ ⟨ proj\_key ⟩

Options:

*delimiter* This option sets the delimiter to use to separate columns.

Valid values: comma  
tab  
userdefined

Default value: **comma**

*userdelimiter*

This option sets the delimiter defined by the user to separate columns.

Valid values: text strings  
Default value:

Operands:

$\langle \text{filename} \rangle \langle \text{import\_key} \rangle \langle \text{proj\_key} \rangle$

The name of the file from which entries will be read. The key to be considered as the reference from file. Property in project that maps to import\_key.

## **entryimportvibration**

Import vibration data for the given entry

Syntax:

**entryimportvibration** *entry*=⟨ text ⟩ ⟨ filename ⟩

Options:

*entry* This is the entry which the vibration data will be attached to.

Valid values: text strings  
Default value:

Operands:

$\langle \text{filename} \rangle$

The name of the file from which vibration data will be imported.

## **entryinvertselection**

Invert the selection state of all project entries.

Syntax:

### **entryinvertselection**

### **entrymovetogroup**

Move the given entries to an existing group.

Syntax:

### **entrymovetogroup <group\_name> <ESL>**

Operands:

<group\_name> <ESL>

An existing group name and a valid ESL expression. Moves the entries that match the ESL description to the given group. The entries are removed from the previous groups.

### **entryrename**

Rename selected project entries.

Syntax:

### **entryrename replace=yes | no <new\_name> <ESL>**

Options:

*replace* This determines whether a single selected entry is allowed to replace an existing entry having the name <new\_name>. Otherwise, the user will be asked to accept a unique name based upon <new\_name>.

Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **false**

Operands:

<new\_name> <ESL>

The new name to be given to a single selected project entry, or the basename for multiple selected project entries from which unique names will be derived

for each. The new name should be followed by a valid ESL expression to specify which entries are to be renamed. If no name is specified, no rename will be done.

## **entryselect**

Select specified entries in current project.

Syntax:

**entryselect** <ESL>

Operands:

<ESL>

A valid ESL specification. Selects those entries which match the ESL description, adding to the currently selected entries.

## **entryselectall**

Select all project entries.

Syntax:

**entryselectall**

## **entryselectonly**

Select only the specified entries in current project.

Syntax:

**entryselectonly** <ESL>

Operands:

<ESL>

A valid ESL specification. Selects only those entries which match the ESL description, unselecting all other entries.

## **entryselectonlyrow**

Select only the entry corresponding to the given row in the current project.

Syntax:

**entryselectonlyrow** <row>

Operands:

<row>

A row number between 1 and the total number of rows. Selects only the entry corresponding to the given row, unselecting all other entries.

## **entryselectrandom**

Selects a random set of entries.

Syntax:

**entryselectrandom** *entries=selected | all* *percentage=<n>* None

Options:

*entries* This determines whether the random entries will be chosen from the selected entries or from all entries.

Valid values: selected

all

Default value: **selected**

*percentage* This is the percentage of entries to select.

Valid values: integers

Default value: **50**

Minimum: 0

Maximum: 100

Operands:

None

None

## **entryselectrow**

Select the entry corresponding to the given row in the current project.

Syntax:

**entryselectrow** ⟨row⟩

Operands:

⟨row⟩

A row number between 1 and the total number of rows. Selects the entry corresponding to the given row.

## **entrysetprop**

Set property value for selected project entries.

Syntax:

**entrysetprop** *property*=⟨text⟩ *value*=⟨text⟩ ⟨ESL⟩

Options:

*property*      The name of the property to be modified. This can be either the user name or the m2io data name for the property. If the property is the entry name (e.g. Entry Name or s\_m\_entry\_name ), an entryrename with replace=no will be done. If the property is the 'included in Workspace' property (In or b\_m\_entry\_in\_workspace ), then entrywsinclude or entrywsexclude will be done, if needed.

Valid values:    text strings

Default value:

*value*      The value to be set for the selected entries. For Boolean properties, legal values are yes, no, true, false, on, off, and 1, 0.

Valid values:    text strings

Default value:

Operands:

⟨ESL⟩

The specified property value will be assigned to the entries selected by the ESL expression.

## **entrysettitle**

Set entry title.

Syntax:

**entrysettitle** <title> <ESL>

Operands:

<title> <ESL>

New entry title and a valid ESL specification. Set title for those entries which match the ESL description from the workspace.

## **entryshowall**

Creates a entry subset in the project table consisting of all the entries in project.

Syntax:

**entryshowall**

## **entrytable**

This keyword is used to set various options associated with the project entry table (Project Table).

Syntax:

**entrytable** *swapnametitle*=yes | no

Options:

*swapnametitle*

This determines whether the table is shown and otherwise treated as if the entry name and title columns are swapped with respect to their positions in the project table.

Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **true**

## **entryunselect**

Unselect specified entries in current project.

Syntax:

**entryunselect** <ESL>

Operands:

<ESL>

A valid ESL specification. Unselect those entries which match the ESL description.

## **entryunselectall**

Unselect all project entries.

Syntax:

**entryunselectall**

## **entryunselectrow**

Unselect the entry corresponding to the given row.

Syntax:

**entryunselectrow** <row>

Operands:

<row>

A row number between 1 and the total number of rows in the project. Unselects the entry corresponding to the given row number.

## **entrywscreate**

Create project entry from atoms in the workspace.

Syntax:

**entrywscreate** *replace=yes | no < new\_name > < ASL >*

Options:

*replace* This option is ignored starting with version 60105 of Maestro.

Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **true**

Operands:

*< new\_name > < ASL >*

The name to be given to the created project entry, followed by a valid ASL expression to specify which atoms constitute the new project entry. If no name is specified, no operation will be done.

## **entrywsexclude**

Exclude selected project entries from the workspace.

Syntax:

**entrywsexclude** *< ESL >*

Operands:

*< ESL >*

A valid ESL specification. Excludes those entries which match the ESL description from the workspace.

## **entrywsexcludenotfixed**

Exclude all unfixed entries from the workspace.

Syntax:

**entrywsexcludenotfixed**

**entrywsinclude**

Include selected project entries into the workspace.

Syntax:

**entrywsinclude** ⟨ESL⟩

Operands:

⟨ESL⟩

A valid ESL specification. Include those entries which match the ESL description into the workspace.

**entrywsincludelock**

Locks the given entries into the workspace.

Syntax:

**entrywsincludelock** ⟨ESL⟩

Operands:

⟨ESL⟩

A valid ESL specification. Locks the given entries into the workspace.

**entrywsincludeonly**

Include only selected project entries in the workspace. Exclude all others.

Syntax:

**entrywsincludeonly** ⟨ESL⟩

Operands:

⟨ESL⟩

A valid ESL specification. Include only those entries which match the ESL description in the workspace. Exclude all others.

## epik

This keyword is used to set various options associated with running epik jobs.

Syntax:

```
epik analysis_mode=query | predict gen_tautomers=yes | no
    input_file=<text> max_output_struct=<n>
    original_ion_state=yes | no original_tautomer=yes | no
    ph=<x> ph_tolerance=<x> pka_file=<text> solvent=h2o |
    dmso structure_source=selected_entries | workspace | file
    tautomer_file=<text>
```

Options:

*analysis\_mode*

In predict mode it generates the new structure and calculates the pka values for the ionizable atoms in these new structures.  
In Querry mode it just calculates pka values for certain ionizable atoms in the structure.

Valid values:    query  
                  predict

Default value: **predict**

*gen\_tautomers*

Generate tautomers

Valid values:    boolean (true|false; yes|no; y|n; on|off)  
Default value: **true**

*input\_file*    The name of the structure input file.

Valid values:    text strings  
Default value:

*max\_output\_struct*

Maximum output structure to be generated.

Valid values:    integers  
Default value: **16**  
Minimum:        0

*original\_ion\_state*

Include original ionization state

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **false**

*original\_tautomer*

Include original Tautomer.

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **false**

*ph*

Ionization pH

Valid values: reals  
Default value: **7**  
Minimum: 0.0  
Maximum: 14.0

*ph\_tolerance*

Ionization pH tolerance

Valid values: reals  
Default value: **2**  
Minimum: 0.0  
Maximum: 7.0

*pka\_file*

The name of the custom pKa parameter file.

Valid values: text strings  
Default value:

*solvent*

Specify the solvent used for calculating pka values for the structure.

Valid values: h2o  
dmso  
Default value: **h2o**

*structure\_source*

Whether to use the selected entries in the current project, or what is in the workspace, or a specified file with multiple structures as structure input for the job.

Valid values: selected\_entries  
workspace  
file  
Default value: **workspace**

*tautomer\_file*

The name of the custom tautomer parameter file.

Valid values: text strings  
Default value:

## **epikstart**

Start a epik job with the current settings.

Syntax:

### **epikstart**

## **eplayergoto**

Go to the specified entry in the ordered sequence of selected project entries and pause there. The table specified in the eplayersettings command determines the sequence order.

Syntax:

**eplayergoto** <entry\_name>

Operands:

<entry\_name>

The name of the entry, within the ordered sequence of selected project entries, which is to be included in the workspace. This frame is recorded as an option to the eplayersettings command.

## **eplayergotofirst**

Go to the first entry in the ordered sequence of selected project entries. The table specified in the eplayersettings command determines the sequence order.

Syntax:

## eplayergotofirst

### eplayergotolast

Go to the last entry in the ordered sequence of selected project entries. The table specified in the eplayersettings command determines the sequence order.

Syntax:

## eplayergotolast

## eplayersettings

Set eplayer state variables.

Syntax:

```
eplayersettings frame=<text> frameduration=<x>
    playmode=loop | reverse | once playsync=yes | no
    referentry=<text> script=noaction | current | file
    scriptfile=<text> superimpose=none | previous | reference
    table=<text> title=<text>
```

Options:

*frame* This option sets the name of the entry being displayed.

Valid values: text strings

Default value:

*frameduration*

This option determines the minimum duration, in seconds, of each displayed (entry) frame during continuous play. The actual frame duration may be longer than the specified value, due to time required for drawing and screen update.

Valid values: reals

Default value: **0**

Minimum: 0.0

Maximum: 5.0

*playmode* This option sets the mode for continuous play. Valid values are “loop”, “reverse”, or “once”. These cause play to continue,

	change direction, or stop, respectively, when reaching either end of the current entry selection.
	Valid values:    loop reverse once Default value: <b>once</b>
<i>playsync</i>	If true, save Workspace changes during continuous play. Valid values:    boolean (true false; yes no; y n; on off) Default value: <b>false</b>
<i>referentry</i>	This option sets the name of the reference entry used when the superimpose option is set to “reference”. The reference entry must exist in the current project in order to use this option. Valid values:    text strings Default value:
<i>script</i>	This option sets the action for each step in the ePlayer. Valid values are “noaction”, “current”, or “file”. At each step, ePlayer may take no action, run current command script, or run command script from a script file. Valid values:    noaction current file Default value: <b>noaction</b>
<i>scriptfile</i>	This option sets the name of the script file used when the script option is set to “Execute Command Script From File”. Valid values:    text strings Default value:
<i>superimpose</i>	This option sets the type of superposition done for entries displayed by the eplayer. Valid values are “none”, “previous”, or “reference”. These cause the incoming entry to be displayed with its current coordinates, with its atoms superimposed upon those of the outgoing entry, or with its atoms superimposed upon those of the reference structure for registration. Valid values:    none previous reference Default value: <b>none</b>
<i>table</i>	This option identifies the project table which is used to define to sequence order of the selected entries in the project. Valid values:    text strings Default value:

*title*      This option sets the title of the entry being displayed.

Valid values:    text strings

Default value:

## **eplayerstepahead**

Go to the next entry in the ordered sequence of selected project entries, if there is one after the frame specified in the eplayersettings command. The table specified in the eplayersettings command determines the sequence order.

Syntax:

### **eplayerstepahead**

## **eplayerstepback**

Go to the previous entry in the ordered sequence of selected project entries, if there is one before the frame specified in the eplayersettings command. The table specified in the eplayersettings command determines the sequence order.

Syntax:

### **eplayerstepback**

## **errorcheck**

Specifies optional error checking to be performed.

Syntax:

**errorcheck** *ct=yes | no*

Options:

*ct*      An option which determines whether mmct error checking is enabled during certain (primarily graphical) operations.

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **false**

## **excludedvolumesmarkersettings**

Set graphical data of Phase excluded volume markers.

Syntax:

```
excludedvolumesmarkersettings ambient=⟨x⟩ diffuse=⟨x⟩  
    emission=⟨x⟩ radius=⟨x⟩ shininess=⟨x⟩ slices=⟨n⟩  
    specular=⟨x⟩ stacks=⟨n⟩ transparency=⟨x⟩
```

Options:

*ambient* Set material property - ambient, to its red, green, and blue components, for front face.

Valid values: reals  
Default value: **0.4**  
Minimum: 0.0  
Maximum: 1.0

*diffuse* Set material property - diffuse, to its red, green, and blue components, for front face.

Valid values: reals  
Default value: **0.4**  
Minimum: 0.0  
Maximum: 1.0

*emission* Set material property - emission, to its red, green, and blue components, for front face.

Valid values: reals  
Default value: **0.1**  
Minimum: 0.0  
Maximum: 1.0

*radius* The radius of QSAR markers.

Valid values: reals  
Default value: **1**  
Minimum: 0.1

*shininess* Set material property - shininess, for front face.

Valid values: reals  
Default value: **80**

	Minimum:	0.0
	Maximum:	128.0
<i>slices</i>	Set the slices of drawing sphere.	
	Valid values: integers	
	Default value: <b>18</b>	
	Minimum: 2	
<i>specular</i>	Set material property - specular, to its red, green, and blue components, for front face.	
	Valid values: reals	
	Default value: <b>0.1</b>	
	Minimum: 0.0	
	Maximum: 1.0	
<i>stacks</i>	Set the stacks of drawing sphere.	
	Valid values: integers	
	Default value: <b>9</b>	
	Minimum: 2	
<i>transparency</i>	The transparency of QSAR markers.	
	Valid values: reals	
	Default value: <b>20</b>	
	Minimum: 0.0	
	Maximum: 100.0	

## **extenddisplaytopose**

This command displays the specified structure from the pose set and any poses between it and the nearest displayed poses before and after it in the list. No poses will be undisplayed (i.e. removed from the workspace).

Syntax:

**extenddisplaytopose** <pose\_name>

Operands:

<pose\_name>

The name of the pose structure.

## fileread

Read a structure file. If a filename is given then structures will be read from that file. If an explicit “start=” option is not included then starting structure is incremented at each read.

For example:

```
fileread start=1 fileread mmodtmp.dat fileread start=1 test.dat
```

Syntax:

```
fileread delete=yes | no format=mmod | pdb | mol2 | maestro |  
sd | jaguarinput | gaussinput | biograf | xyz | jaguaroutput |  
gaussian92 | gaussian94 | gamess | mopaccartesian |  
mopacinternal | mopacoutput | babelpdb | mdl | babelmol |  
babelmol2 | spartan | spartansemi | spartanmm | gamessinput |  
gausscartesian | gaussianz | jaguarzmatrix | jaguarcartesian |  
any | reagentprep ginfo=yes | no start=<n> tile=yes | no  
total=<n> [<file_name>]
```

Options:

*delete* This determines whether the currently displayed structure will be deleted before the new structure is read in.

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **true**

*format* This option sets the format of the file to be read. Valid values are “maestro”, “mmod”, “pdb”, “mol2”, or “sd”.

Valid values:    mmod  
                  pdb  
                  mol2  
                  maestro  
                  sd  
                  jaguarinput  
                  gaussinput  
                  biograf  
                  xyz  
                  jaguaroutput  
                  gaussian92  
                  gaussian94  
                  gamess  
                  mopaccartesian  
                  mopacinternal  
                  mopacoutput  
                  babelpdb  
                  mdl  
                  babelmol  
                  babelmol2  
                  spartan  
                  spartansemi  
                  spartanmm  
                  gamessinput  
                  gausscartesian  
                  gaussianz  
                  jaguarzmatrix  
                  jaguarcartesian  
                  any  
                  reagentprep  
Default value: **maestro**

*ginfo*        This determines whether the file will be read with just the geometric information or if the visual information in the file will be read and used for displaying the structures.

Valid values:    boolean (true|false; yes|no; y|n; on|off)  
Default value: **false**

*start*        This option sets the number of the first structure to be read

Valid values:    integers  
Default value: **1**  
Minimum:        1

*tile*         This determines whether the structures displayed on screen will be “tiled”.

Valid values:    boolean (true|false; yes|no; y|n; on|off)

Default value: **false**

*total* The total number of structures to be read from the file

Valid values: integers

Default value: **1**

Minimum: 1

Operands:

[⟨ file\_name ⟩]

The name of the file from which the structure will be read. If no name is specified, then no file read will be done.

Aliases:

**read** (see [read], page 410)

## filewrite

If the write command is issued with options, but without a filename then nothing is written but the options are updated. If the write command alone is issued then writing is performed to the currently open file with the current options. A write command with both options and a file name specified will result in the write being performed with the new options to the specified file.

Syntax:

```
filewrite append=yes | no displayed_atoms=yes | no
format=mmod | pdb | mol2 | maestro | sd | jaguarinput |
gaussinput | biograf | xyz | jaguaroutput | gaussian92 |
gaussian94 | gamess | mopaccartesian | mopacinternal |
mopacoutput | babelpdb | mdl | babelmol | babelmol2 |
spartan | spartansemi | spartanmm | gamessinput |
gausscartesian | gaussianz | jaguarzmatrix | jaguarcartesian |
any | reagentprep graphical=yes | no separate=yes | no
title=⟨ text ⟩ ⟨ file_name ⟩
```

Options:

*append* This option determines whether to append to the file which is going to be written

Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **false**

*displayed\_atoms*

This option determines whether the displayed atoms will be saved. This option is not currently supported.

Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **true**

*format*

This option sets the format of the file to be read. Valid values are “mmod”, “pdb” or “mol2”.

Valid values:

mmod  
pdb  
mol2  
maestro  
sd  
jaguarinput  
gaussinput  
biograf  
xyz  
jaguaroutput  
gaussian92  
gaussian94  
gameSS  
mopaccartesian  
mopacinternal  
mopacoutput  
babelpdb  
mdl  
babelmol  
babelmol2  
spartan  
spartansemi  
spartanmm  
gameSSinput  
gausscartesian  
gaussianz  
jaguarzmatrix  
jaguarcartesian  
any  
reagentprep

Default value: **maestro**

*graphical*

This option determines whether to write out graphical information when a structure is being saved to disk.

Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **true**

**separate** This option determines whether to write out the CT as one structure or separate structures. This option is not currently supported.

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **false**

**title** The title for the output structure.

Valid values: text strings  
Default value:

Operands:

**⟨file\_name⟩**

The name of the file to which the structure will be written.

Aliases:

**write** (see [[write](#)], page 501)

## **filter**

Creates a new entry filter. The filter name must be a single token (or “quoted” if multiple tokens). A filter can be redefined by specifying a new definition.

Syntax:

**filter** ⟨filter\_name⟩ ⟨ESL-definition⟩

Operands:

**⟨filter\_name⟩** ⟨ESL-definition⟩

The name which will be applied to the filter. If the name contains embedded spaces then it must be enclosed in double quotation marks.

## **find**

Find atom(s), residue(s), chain, or molecule described by the ASL operand.

Syntax:

```
find atombyname=<text> atombynum=<n> byasl=<text>
    center=yes | no chainname=<text> findmethod=bynumber |
    byname findtype=atom | residue | chain | molecule | asl
    hidemarkers=yes | no inscode=<text> label=yes | no
    markall=yes | no matchesnum=<text> molnum=<n>
    resnum=<n> < ASL >
```

Options:

*atombyname*

This option determines name of the atom to be found.

Valid values: text strings

Default value:

*atombynum*

This option determines value of the atom number to be found

Valid values: integers

Default value: **1**

Minimum: 1

*byasl*

This option determines the ASL specification fo the objects to be found (atom, residue, chain ot molecule).

Valid values: text strings

Default value:

*center*

This option determines whether the structure will center on the found atom or on the centroid of the atom(s) [of residue, chain or molecule].

Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **true**

*chainname*

This option determines the chain name to be found, or the chain name in which to find a specified residue.

Valid values: text strings

Default value:

*findmethod*

This option determines the method used to specify an atom to be found, including By Number and By Name .

Valid values: bynumber

byname

Default value: **bynunber**

*findtype*

This option determines the find type, including five options of atom, residue, chain, molecule and asl.

Valid values: atom  
residue  
chain  
molecule  
asl  
Default value: **atom**

*hidemarkers*

This option determines whether all the markers should be hidden (TRUE) or not (FALSE)

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **true**

*inscode* This option determines value of the insertion code of the residue to be found.

Valid values: text strings  
Default value:

*label* This option determines whether the found atoms will be labeled.

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **true**

*markall* This option determines whether all the found atom(s) or atom(s) of residue, chain or molecule are marked (TRUE), or only the principal atom(s) are marked (FALSE).

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **false**

*matchesnum*

This option shows the number of matched atoms over the total number of atoms in the structure.

Valid values: text strings  
Default value: /

*molnum* This option determines the number of the molecule to be found

Valid values: integers  
Default value: **1**  
Minimum: 1

*resnum* This option determines value of the residue number to be found.

Valid values: integers  
Default value: **1**

Operands:

$\langle \text{ASL} \rangle$

A string in the atom specification language which describes the set of atoms which are to be found.

## **fit**

Fit structures to the workspace.

Syntax:

**fit** ⟨ ASL ⟩

Operands:

⟨ ASL ⟩

If present, we do fit to screen on ASL, if not present we do fit on all atoms.

## **forcefield**

Used to display the force field in the force field viewer.

Syntax:

**forcefield**

## **forcefieldbend**

A keyword which controls the display of the bend interactions in the FF viewer.

Syntax:

**forcefieldbend** *select=⟨ n ⟩ show=all | high | medium | low sortbyenergy=yes | no*

Options:

*select* Set the interaction which is currently selected

Valid values: integers

Default value: **0**

Minimum: 0

*show* Sets a filter which determines which interactions are shown depending on the quality of their parameters.

Valid values: all  
high  
medium  
low  
Default value: **all**

*sortbyenergy*

Determines if the interactions are sorted by energy

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **false**

## **forcefieldele**

A keyword which controls the display of the electrostatic interactions in the FF viewer.

Syntax:

**forcefieldele** *select=<n>* *show=all | high | medium | low*  
*sortbyenergy=yes | no*

Options:

*select* Set the interaction which is currently selected

Valid values: integers  
Default value: **0**  
Minimum: 0

*show* Sets a filter which determines which interactions are shown depending on the quality of their parameters.

Valid values: all  
high  
medium  
low  
Default value: **all**

*sortbyenergy*

Determines if the interactions are sorted by energy

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **false**

## **forcefieldgbsolv**

A keyword which controls the display of the GBSOLV interactions in the FF viewer.

Syntax:

```
forcefieldgbsolv select=<n> show=all | high | medium | low
                    sortbyenergy=yes | no
```

Options:

*select* Set the interaction which is currently selected

Valid values: integers

Default value: **0**

Minimum: 0

*show* Sets a filter which determines which interactions are shown depending on the quality of their parameters.

Valid values: all  
high  
medium  
low

Default value: **all**

*sortbyenergy*

Determines if the interactions are sorted by energy

Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **false**

## **forcefieldimproper**

A keyword which controls the display of the improper torsion interactions in the FF viewer.

Syntax:

```
forcefieldimproper select=<n> show=all | high | medium | low
                     sortbyenergy=yes | no
```

Options:

*select* Set the interaction which is currently selected

Valid values: integers  
Default value: **0**  
Minimum: 0

*show* Sets a filter which determines which interactions are shown depending on the quality of their parameters.  
Valid values: all  
high  
medium  
low  
Default value: **all**

*sortbyenergy*  
Determines if the interactions are sorted by energy  
Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **false**

## **forcefieldsasolv**

A keyword which controls the display of the SASOLV interactions in the FF viewer.

Syntax:

**forcefieldsasolv** *select=*< n > *show=*all | high | medium | low  
*sortbyenergy=*yes | no

Options:

*select* Set the interaction which is currently selected  
Valid values: integers  
Default value: **0**  
Minimum: 0

*show* Sets a filter which determines which interactions are shown depending on the quality of their parameters.  
Valid values: all  
high  
medium  
low  
Default value: **all**

*sortbyenergy*  
Determines if the interactions are sorted by energy

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **false**

## forcefieldstretch

A keyword which controls the display of the stretch interactions in the FF viewer.

Syntax:

**forcefieldstretch** *select*=⟨n⟩ *show*=all | high | medium | low  
*sortbyenergy*=yes | no

Options:

*select* Set the interaction which is currently selected

Valid values: integers

Default value: **0**

Minimum: 0

*show* Sets a filter which determines which interactions are shown depending on the quality of their parameters.

Valid values: all  
high  
medium  
low

Default value: **all**

*sortbyenergy*

Determines if the interactions are sorted by energy

Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **false**

## forcefieldtorsion

A keyword which controls the display of the torsion interactions in the FF viewer.

Syntax:

**forcefieldtorsion** *select=<n> show=all | high | medium | low  
sortbyenergy=yes | no*

Options:

*select* Set the interaction which is currently selected

Valid values: integers

Default value: **0**

Minimum: 0

*show* Sets a filter which determines which interactions are shown depending on the quality of their parameters.

Valid values: all  
high  
medium  
low

Default value: **all**

*sortbyenergy*

Determines if the interactions are sorted by energy

Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **false**

## forcefieldvdw

A keyword which controls the display of the VDW interactions in the FF viewer.

Syntax:

**forcefieldvdw** *select=<n> show=all | high | medium | low  
sortbyenergy=yes | no*

Options:

*select* Set the interaction which is currently selected

Valid values: integers

Default value: **0**

Minimum: 0

*show* Sets a filter which determines which interactions are shown depending on the quality of their parameters.

Valid values: all  
high  
medium  
low  
Default value: **all**

*sortbyenergy*

Determines if the interactions are sorted by energy

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **false**

## forcefieldview

Used to specify a .mmo file which contains the interactions to be displayed in the FF viewer.

Syntax:

**forcefieldview** <mmo\_file\_name>

Operands:

<mmo\_file\_name>

The name of the .mmo file which is to be opened and displayed. The full filename, including the .mmo suffix must be specified.

## forcefieldwilson

A keyword which controls the display of the wilson interactions in the FF viewer.

Syntax:

**forcefieldwilson** *select=*<n> *show=*all | high | medium | low  
*sortbyenergy=*yes | no

Options:

*select* Set the interaction which is currently selected  
Valid values: integers  
Default value: **0**  
Minimum: 0

**show** Sets a filter which determines which interactions are shown depending on the quality of their parameters.

Valid values:    all  
                  high  
                  medium  
                  low

Default value: **all**

**sortbyenergy**

Determines if the interactions are sorted by energy

Valid values:    boolean (true|false; yes|no; y|n; on|off)  
Default value: **false**

## **formalcharge**

Increment, decrement or set the formal charge for atoms which match the ASL specification.

Syntax:

**formalcharge** increment|decrement|⟨ formal\_charge ⟩ ⟨ ASL ⟩

Operands:

increment|decrement|⟨ formal\_charge ⟩ ⟨ ASL ⟩

The first operand is either the word “increment”, the word “decrement”, or an integer value representing the formal charge to be used for all atoms which match the specification. “increment” and “decrement” change the atom’s current formal charge by +1 or -1 respectively. The second operand is a valid ASL string which defines the set of atoms which are to have their atom names changed. An error will be issued for the first atom in the set which cannot have its formal charge set (usually because no suitable charged type exists) and no further atoms will have their charge changed.

## **fragment**

This command sets the fragment mode and the current fragment used for growing, placing and replacing.

Syntax:

**fragment** ⟨ fragment\_mode ⟩ [⟨ current\_fragment ⟩]

Operands:

⟨ fragment\_mode ⟩ [⟨ current\_fragment ⟩]

The first operand is the name of the fragment mode, this must be a valid name from the file \$SCHRODINGER/mmshare-vX.X/data/res/mmfrag.ini. The optional second argument is the name of a fragment within that mode which is to be made the current fragment.

## **fragmenttype**

This command sets the fragment type for the current fragment mode.

Syntax:

**fragmenttype** ⟨ fragment\_type ⟩

Operands:

⟨ fragment\_type ⟩

The first operand is the a qualifier for the fragment which some modes (such as furanose and pyranose) have. For example alpha or beta for pyranose or furanose sugars. The fragment\_type must be a valid type for the current fragment mode.

## **frozenatom**

Specifies a single atom to be frozen at its current position in a MacroModel calculation. Other atoms in the molecule will continue to interact with this atom. The frozen atom however will not “feel” any other atoms in the molecule.

Syntax:

**frozenatom** ⟨ atom\_number ⟩

Operands:

⟨ atom\_number ⟩

The number of an atom which is to be treated as frozen during a MacroModel calculation.

## **frozensest**

Specifies a set of atoms to be frozen at their current positions in a Macro-Model calculation. Other atoms in the molecule will continue to interact with these atoms. The frozen atoms however will not “feel” any other atoms in the molecule.

Syntax:

**frozensest** ⟨ ASL ⟩

Operands:

⟨ ASL ⟩

A string in the atom specification language which describes the set of atoms which are to be treated as frozen in a MacroModel calculation.

## **fuse**

Fuse atom list pairs

Syntax:

**fuse**

## **geometrycleanup**

Cleans up the geometry

Syntax:

## **geometrycleanup**

### **glideactivesiteres**

Defines a residue to be included in the active site for a Glide grid calculation.

Syntax:

**glideactivesiteres** <chain>:<molnum>:<resnum>:<insertioncode>

Operands:

<chain>:<molnum>:<resnum>:<insertioncode>

A colon separated list giving the PDB chain name, the residue number and insertion code for a residue to be included in the list of active site residues.

### **glideactivesiteset**

Defines a set from atoms, from which all residues will be added as part of the active site for Glide calculations.

Syntax:

**glideactivesiteset** <ASL>

Operands:

<ASL>

A valid ASL specification. All residues which have any atoms in this set will be added to the list of residues that define the active site.

### **glidecalcboxfromligand**

Uses the information specified in the glideligandgrid command to calculate the grid box for a Glide calculation.

Syntax:

## **glidecalcboxfromligand**

### **glideconstraintatomlabel**

Specifies a constraint atom label in the receptor for a Glide calculation.

Syntax:

**glideconstraintatomlabel** <atom-number> <label>

Operands:

<atom-number> <label>

The name which will be applied to the atom.

### **glideconstraintposition**

Specifies a constraint position in the receptor for a Glide calculation.

Syntax:

**glideconstraintposition** *radius*=<x> *x*=<x> *y*=<x> *z*=<x>

Options:

*radius*      The radius of a Glide constraint position.

Valid values:    reals

Default value:    **1**

Minimum:        0.0001

*x*      The X coordinate of a Glide constraint position.

Valid values:    reals

Default value:    **0**

*y*      The Y coordinate of a Glide constraint position.

Valid values:    reals

Default value:    **0**

*z*      The Z coordinate of a Glide constraint position.

Valid values:    reals

Default value:    **0**

## **glideconstraintregion**

Specifies a constraint region in the receptor for a Glide calculation.

Syntax:

**glideconstraintregion** <region\_name> <cell>

Operands:

<region\_name> <cell>

The name which will be applied to the region. If the name contains embedded spaces then it must be enclosed in double quotation marks.

## **glideconstraintregionactivecell**

Set an active cell for the region.

Syntax:

**glideconstraintregionactivecell** <region\_name> <active\_cell>

Operands:

<region\_name> <active\_cell>

The name which will be applied to the region. If the name contains embedded spaces then it must be enclosed in double quotation marks.

## **glideconstraintregiongrow**

Grow cells for a region.

Syntax:

**glideconstraintregiongrow** <region\_name>

Operands:

<region\_name>

The name which will be applied to the region. If the name contains embedded spaces then it must be enclosed in double quotation marks.

## **glideconstraintregioninvisible**

Set a constraint region as invisible in workspace.

Syntax:

**glideconstraintregioninvisible** <region\_name>

Operands:

<region\_name>

The name which will be applied to the region. If the name contains embedded spaces then it must be enclosed in double quotation marks.

## **glideconstraintregionshrink**

Shrink cells for a region.

Syntax:

**glideconstraintregionshrink** <region\_name>

Operands:

<region\_name>

The name which will be applied to the region. If the name contains embedded spaces then it must be enclosed in double quotation marks.

## **glideconstraintregionvisible**

Set a constraint region as visible in workspace.

Syntax:

**glideconstraintregionvisible** <region\_name>

Operands:

<region\_name>

The name which will be applied to the region. If the name contains embedded spaces then it must be enclosed in double quotation marks.

**glidedisplayreceptor**

Displays the receptor in the Workspace, clearing any existing scratch entries.

Syntax:

**glidedisplayreceptor****glidedockconstraintposition**

Specifies a constraint position in the receptor for a Glide calculation.

Syntax:

```
glidedockconstraintposition feature=⟨n⟩ index=⟨n⟩
    radius=⟨x⟩ type=⟨n⟩ use1=yes | no use2=yes | no
    use3=yes | no use4=yes | no x=⟨x⟩ y=⟨x⟩ z=⟨x⟩
```

Options:

*feature*      The constraint feature of position in docking.

Valid values:    integers

Default value:    **-1**

*index*      The index of position in docking.

Valid values:    integers

Default value:    **0**

*radius*      The radius of a Glide constraint position.

Valid values:    reals

Default value:    **1**

Minimum:        0.0001

*type*      The constraint type of position in docking.

Valid values:    integers

Default value:    **0**

*use1*      The flag indicates if this position will be used in docking for group 1.

Valid values:    boolean (true|false; yes|no; y|n; on|off)

Default value:    **false**

*use2*      The flag indicates if this position will be used in docking for group 2.

	Valid values: boolean (true false; yes no; y n; on off) Default value: <b>false</b>
<i>use3</i>	The flag indicates if this position will be used in docking for group 3. Valid values: boolean (true false; yes no; y n; on off) Default value: <b>false</b>
<i>use4</i>	The flag indicates if this position will be used in docking for group 4. Valid values: boolean (true false; yes no; y n; on off) Default value: <b>false</b>
<i>x</i>	The X coordinate of a Glide constraint position. Valid values: reals Default value: <b>0</b>
<i>y</i>	The Y coordinate of a Glide constraint position. Valid values: reals Default value: <b>0</b>
<i>z</i>	The Z coordinate of a Glide constraint position. Valid values: reals Default value: <b>0</b>

## glidedockconstraintregion

Specifies a constraint region in the receptor for a Glide calculation.

Syntax:

```
glidedockconstraintregion atoms=<n> feature=<n> index=<n>
    type=<n> use1=yes | no use2=yes | no use3=yes | no
    use4=yes | no <region_name>
```

Options:

<i>atoms</i>	The number of required ligand atoms of constraint region in docking. Valid values: integers Default value: <b>1</b>
<i>feature</i>	The constraint feature of constraint region in docking. Valid values: integers Default value: <b>-1</b>

<i>index</i>	The index of constraint region in docking. Valid values: integers Default value: <b>0</b>
<i>type</i>	The constraint type of constraint region in docking. Valid values: integers Default value: <b>0</b>
<i>use1</i>	The flag indicates if this constraint region will be used in docking for group 1. Valid values: boolean (true false; yes no; y n; on off) Default value: <b>false</b>
<i>use2</i>	The flag indicates if this constraint region will be used in docking for group 2. Valid values: boolean (true false; yes no; y n; on off) Default value: <b>false</b>
<i>use3</i>	The flag indicates if this constraint region will be used in docking for group 3. Valid values: boolean (true false; yes no; y n; on off) Default value: <b>false</b>
<i>use4</i>	The flag indicates if this constraint region will be used in docking for group 4. Valid values: boolean (true false; yes no; y n; on off) Default value: <b>false</b>

Operands:

*<region\_name>*

The name which will be applied to the region. If the name contains embedded spaces then it must be enclosed in double quotation marks.

## **glidedockregionnumatoms**

Specifies the required ligand atoms for this constraint region.

Syntax:

**glidedockregionnumatoms** *<region\_name>* *<num\_atoms>*

Operands:

*<region\_name>* *<num\_atoms>*

The name of the region and the minimum number of atoms permitted for this region. If the name contains embedded spaces then it must be enclosed in double quotation marks.

## **glidedockregionselect**

Selects a hydrophobic constraint region to be used as a constraint in the docking job.

Syntax:

**glidedockregionselect** *group=* $\langle n \rangle$  *<region\_name>*

Options:

*group*      The constraint group for region.  
Valid values:    integers  
Default value:   1  
Minimum:        0  
Maximum:       3

Operands:

*<region\_name>*

The name of the region to be selected as a constraint in the docking job. If the name contains embedded spaces then it must be enclosed in double quotation marks.

## **glidedockregionunselect**

Unselects a hydrophobic constraint region so it will not be used as a constraint in the docking job.

Syntax:

**glidedockregionunselect** *group=* $\langle n \rangle$  *<region\_name>*

Options:

*group*      The constraint group for region.  
Valid values:    integers

Default value: **1**  
Minimum: 0  
Maximum: 3

Operands:

$\langle \text{region\_name} \rangle$

The name of the region to be unselected as a constraint in the docking job. If the name contains embedded spaces then it must be enclosed in double quotation marks.

## glidegridhydrophobic

Used to set all the options associated with the hydrophobic constraints in Glide grid generation.

Syntax:

**glidegridhydrophobic** *num\_vertices*= $\langle n \rangle$  *threshold*= $\langle x \rangle$

Options:

*num\_vertices*

At least this number of cell vertices must meet the threshold in order for the cell to be displayed.

Valid values: integers  
Default value: **5**  
Minimum: 1  
Maximum: 8

*threshold* This is cutoff for which grid cells are considered hydrophobic.

Valid values: reals  
Default value: **-0.5**  
Minimum: -0.65  
Maximum: -0.01

## glideligand

Specifies settings about the ligand(s) to be used in a Glide job.

Syntax:

```
glideligand amidebondrotations=yes | no baddist=<x>
    bondrotation=forbid | allow conformations=usesupplied |
    generate | inplace definereference=yes | no
    dockdisplayed=yes | no dockfromfile=yes | no dockrange=all |
    specified econfcut=<x> endlig=<n> format=maestro | sd |
    mol2 | pdb lig_ccut=<x> lig_vscale=<x> ligandsfile=<text>
    ligandsource=neither | entries | extfile reflig=yes | no
    ringconf=yes | no startlig=<n>
```

Options:

*amidebondrotations*

An option which allows amide bond rotations.

Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **true**

*baddist* The distance criterion for bad internal contacts

Valid values: reals

Default value: **2.45**

Minimum: 0.0

*bondrotation*

The option of amide bond rotation.

Valid values: forbid

allow

Default value: **allow**

*conformations*

Whether to use only supplied conformations, to generate them with confgen or to treat them “in-place”.

Valid values: usesupplied

generate

inplace

Default value: **generate**

*definereference*

An option which determines if a reference ligand is to be defined.

Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **false**

*dockdisplayed*

An option which determines if the currently displayed ligand will be docked.

Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **false**

*dockfromfile*

[NOTE: This option is no longer used.] An option which determines ligands from external files are to be docked:

Valid values: boolean (true|false; yes|no; y|n; on|off)  
 Default value: **true**

*dockrange* The range of structures in the file to be docked - whether to dock all from the file or a specified range.

Valid values: all  
 specified  
 Default value: **all**

*econfcut* The energy window for keeping conformations.

Valid values: reals  
 Default value: **12**  
 Minimum: 0.0

*endlig* The final structure from the file to be docked.

Valid values: integers  
 Default value: **1000**  
 Minimum: 0

*format* The format of the ligands.

Valid values: maestro  
 sd  
 mol2  
 pdb  
 Default value: **maestro**

*lig\_ccut* The partial atomic charge below which ligand atoms are considered to be non-polar and will have their VDW radii scaled.

Valid values: reals  
 Default value: **0.15**  
 Minimum: 0.00000001

*lig\_vscale* The scaling factor for the VDW radii of non-polar ligand atoms.

Valid values: reals  
 Default value: **0.8**  
 Minimum: 0.00000001

*ligandsfile* The file containing one or more ligands.

Valid values: text strings  
 Default value:

*ligandsource*

The source of the ligands to be docked

	Valid values:	neither entries extfile Default value: <b>extfile</b>
<i>reflig</i>	An option which determines if a reference ligand is to be used in this Glide job.	
	Valid values:	boolean (true false; yes no; y n; on off)
	Default value:	<b>false</b>
<i>ringconf</i>	An option which allows ring flips	
	Valid values:	boolean (true false; yes no; y n; on off)
	Default value:	<b>true</b>
<i>startlig</i>	The first structure from the file to be docked.	
	Valid values:	integers
	Default value:	<b>1</b>
	Minimum:	1

## **glideligandfile**

Specifies a file containing one or more ligands to be used for a Glide docking run.

Syntax:

**glideligandfile** <ligand\_file\_name>

Operands:

<ligand\_file\_name>

The name of a file containing one or more ligands.

## **glideligandgrid**

Defines the ligand to be used to calculate the grid for the glide calculation.

Syntax:

**glideligandgrid** *ligandentry*=⟨text⟩ *ligandformat*=maestro | sd | mol2 | pdb *ligandindex*=⟨n⟩ *ligandsource*=displayed | entryname | extfile | none *usedisplayed*=yes | no  
⟨ligand file name⟩

Options:

*ligandentry*

The name of the entry that will be used to define the grid for the Glide calculation.

Valid values: text strings

Default value:

*ligandformat*

The format of the file that will be used to calculate the grid box.

Valid values: maestro  
sd  
mol2  
pdb

Default value: **maestro**

*ligandindex*

The index of the structure that is to be read as a ligand.

Valid values: integers

Default value: **1**

Minimum: 1

*ligandsource*

The source of the ligand to be used to define the Grid for the Glide calculation.

Valid values: displayed  
entryname  
extfile  
none

Default value: **displayed**

*usedisplayed*

[NOTE: This option is no longer used.] An option which determines if the currently displayed ligand will be used to calculate the grid for the Glide calculation

Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **false**

Operands:

⟨ligand file name⟩

The name of the file containing the ligand to be used to define the grid.

## glidelocatehydrophobiccells

Runs hppmap and displays hydrophobic cells.

Syntax:

## glidelocatehydrophobiccells

## glideoutput

Controls a number of options for how the poses are output during a Glide calculation.

Syntax:

```
glideoutput delpose=⟨x⟩ maxperlig=⟨n⟩ nreport=⟨n⟩  
          outputdisp=poseviewer | ligandsonly posedist=⟨x⟩
```

Options:

*delpose* The maximum displacement for discarding duplicate poses.

Valid values: reals

Default value: **1.3**

Minimum: 0.0

*maxperlig* The maximum number of poses per ligand to be written.

Valid values: integers

Default value: **1**

Minimum: 0

*nreport* The number of coordinate sets to write to disk.

Valid values: integers

Default value: **10000**

Minimum: 1

*outputdisp* Whether the output structure file will be intended to be viewed with the Maestro pose viewer (and include the receptor structure) or is just to be the docked ligands.

Valid values: poseviewer

ligandsonly

Default value: **poseviewer**

*posedist* The distance criterion for distinct poses.

Valid values: reals

Default value: **0.5**  
Minimum: 0.0

## glidereceptorligand

Defines a on-screen molecule to be treated as the ligand for a Glide calculation

Syntax:

**glidereceptorligand** < molecule\_number >

Operands:

< molecule\_number >

The number of a molecule to be included as the ligand.

## gliderefenceligand

Specifies a file containing the reference ligand for a glide docking run.

Syntax:

**gliderefenceligand** *index=*< n > *ligandentry=*< text >  
*ligandfile=*< text > *ligandformat=*maestro | sd | mol2 | pdb  
*ligandsource=*displayed | entryname | extfile | none  
*use\_displayed=*yes | no

Options:

*index* The index of the reference ligand in the reference ligand file.  
Valid values: integers  
Default value: **1**  
Minimum: 1

*ligandentry*  
The name of the entry that will be used as the reference ligand.  
Valid values: text strings  
Default value:

*ligandfile* The name of the file containing the ligand structure that will be used as the reference ligand.

Valid values: text strings

Default value:

*ligandformat*

The format of the file that will be used to calculate the grid box.

Valid values: maestro

sd

mol2

pdb

Default value: **maestro**

*ligandsource*

The source of the ligand to be used as the reference ligand.

Valid values: displayed

entryname

extfile

none

Default value: **none**

*use\_displayed*

[NOTE: This option is no longer used.] An option which determines if the reference ligand is that which is already displayed.

Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **false**

## glidescoring

Controls a number of options for how the poses are scored.

Syntax:

```
glidescoring cvcutoff=<x> dielectric=<x> hbcutoff=<x>
    itmax=<n> maxkeep=<n> maxref=<n> mlcutoff=<x>
    posedist=<x> scorecut=<x> short_distance=anneal | softcore
```

Options:

*cvcutoff* Reject poses if the Coulomb/VDW energy exceeds this value

Valid values: reals

Default value: **0**

*dielectric* The dielectric constant used in the grid energy calculation.

Valid values: reals

Default value: **2**

Minimum: 0.999999999

<i>hbcutoff</i>	Reject poses if the H-bond energy exceeds this value Valid values:   reals Default value: <b>0</b> Maximum:        0.00000001
<i>itmax</i>	The maximum number of conjugate gradient steps Valid values:   integers Default value: <b>100</b> Minimum:        0
<i>maxkeep</i>	The number of ligand poses on the coarse grid. Valid values:   integers Default value: <b>5000</b> Minimum:        1
<i>maxref</i>	The maximum number of poses to keep after rough score refinement. Valid values:   integers Default value: <b>400</b> Minimum:        1
<i>mlcutoff</i>	Reject poses if the metal-ligand interaction score exceeds this value. Valid values:   reals Default value: <b>0</b> Maximum:        0.00000001
<i>posedist</i>	The distance criterion for distinct poses. Valid values:   reals Default value: <b>0.5</b> Minimum:        0.0
<i>scorecut</i>	The window of rough scores for keeping poses. Valid values:   reals Default value: <b>100</b> Minimum:        0.0
<i>short_distance</i>	The short distance behavior of the potentials. Valid values:   anneal softcore Default value: <b>anneal</b>

## glidesettings

Allows the settings of some values that determine how the overall Glide job runs.

Syntax:

```
glidesettings function=run | setup griddirectory=<text>
    gridfilename=<text> maxatom=<n> maxrotbonds=<n>
    mode=normal | throughput | accurate numjobs=<n>
    numprocs=<n> numreqgroup1=<n> numreqgroup2=<n>
    numreqgroup3=<n> numreqgroup4=<n> receptor=alone |
    withligand | ligandonly recyclesteps=<n> reqmodegroup1=all |
    atleast reqmodegroup2=all | atleast reqmodegroup3=all |
    atleast reqmodegroup4=all | atleast source=structure | file
    splitligands=yes | no writegrid=yes | no
```

Options:

*function* Determines whether the Glide job is actually to dock some ligands or just calculate a grid for a receptor.

Valid values: run  
setup  
Default value: **setup**

*griddirectory*

The directory where the grid files are located.

Valid values: text strings  
Default value:

*gridfilename*

The base name for the file which the receptor grid is to be written to or read from.

Valid values: text strings  
Default value:

*maxatom*

Any ligands in the input with more than this number of atoms will be skipped.

Valid values: integers  
Default value: **120**  
Minimum: 1  
Maximum: 200

*maxrotbonds*

Any ligands in the input with more than this number of rotatable bonds will be skipped.

	Valid values:	integers
	Default value:	<b>20</b>
	Minimum:	1
<i>mode</i>	Determines how a number of other values are set to ensure the job runs either faster than normal or more accurate than normal.	
	Valid values:	normal throughput accurate
	Default value:	<b>normal</b>
<i>numjobs</i>	Number of jobs to split this docking job into	
	Valid values:	integers
	Default value:	<b>1</b>
	Minimum:	1
<i>numprocs</i>	Number of processors to use for each job.	
	Valid values:	integers
	Default value:	<b>1</b>
	Minimum:	1
<i>numreqgroup1</i>	Number of constraints to be required for group 1 in docking.	
	Valid values:	integers
	Default value:	<b>1</b>
	Minimum:	0
	Maximum:	4
<i>numreqgroup2</i>	Number of constraints to be required for group 2 in docking.	
	Valid values:	integers
	Default value:	<b>1</b>
	Minimum:	0
	Maximum:	4
<i>numreqgroup3</i>	Number of constraints to be required for group 3 in docking.	
	Valid values:	integers
	Default value:	<b>1</b>
	Minimum:	0
	Maximum:	4
<i>numreqgroup4</i>	Number of constraints to be required for group 4 in docking.	
	Valid values:	integers
	Default value:	<b>1</b>
	Minimum:	0
	Maximum:	4

*receptor*      Determines whether the structure displayed is just the receptor or a receptor + ligand.

Valid values:      alone  
                      withligand  
                      ligandonly

Default value:      **alone**

*recyclesteps*

Specifies the number of ligand recycling iterations in high accuracy docking jobs (glidesettings mode=accurate).

Valid values:      integers

Default value:      **5**

Minimum:      1

*reqmodegroup1*

The mode determines how to set the number of required constraints for group 1 in docking.

Valid values:      all  
                      atleast

Default value:      **atleast**

*reqmodegroup2*

The mode determines how to set the number of required constraints for group 2 in docking.

Valid values:      all  
                      atleast

Default value:      **atleast**

*reqmodegroup3*

The mode determines how to set the number of required constraints for group 3 in docking.

Valid values:      all  
                      atleast

Default value:      **atleast**

*reqmodegroup4*

The mode determines how to set the number of required constraints for group 4 in docking.

Valid values:      all  
                      atleast

Default value:      **atleast**

*source*      Where the grid files come from: calculated from the structure or from previously calculated files.

Valid values:      structure  
                      file

Default value:      **structure**

*splitligands*

Whether to split the ligand structures into separate files per job  
 Valid values: boolean (true|false; yes|no; y|n; on|off)  
 Default value: **false**

*writegrid* An option which determines if the grid will be written following the job.  
 Valid values: boolean (true|false; yes|no; y|n; on|off)  
 Default value: **false**

**glidesite**

Settings for the site in a glide job.

Syntax:

```
glidesite boxcenter=ligand | residues | supplied boxsize=⟨n⟩
boxsizeX=⟨n⟩ boxsizeY=⟨n⟩ boxsizeZ=⟨n⟩
display_center_box=yes | no display_enclosing_box=yes | no
enclosingbox=fitlargest | fitdisplayed | fitlength
maxliglength=⟨x⟩ recep_ccut=⟨x⟩ recep_vscaled=⟨x⟩
xcent=⟨x⟩ ycent=⟨x⟩ zcent=⟨x⟩
```

Options:

*boxcenter* Specifies how the box center is defined - by specifying residues in the active site, by specifying a ligand or by using the values as input by the user.

Valid values: ligand  
 residues  
 supplied

Default value: **ligand**

*boxsize* The size of the enclosing box.

Valid values: integers  
 Default value: **10**  
 Minimum: 6  
 Maximum: 40

*boxsizeX* The length in X of the inner box.

Valid values: integers  
 Default value: **10**  
 Minimum: 6  
 Maximum: 40

*boxsizey* The length in Y of the inner box.

Valid values: integers

Default value: **10**

Minimum: 6

Maximum: 40

*boxsizez* The length in Z of the inner box.

Valid values: integers

Default value: **10**

Minimum: 6

Maximum: 40

*display\_center\_box*

An option which determines if the box which contains the ligand center of motion will be displayed.

Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **true**

*display\_enclosing\_box*

An option which determines if the box which encloses the ligand center of motion box will be displayed.

Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **true**

*enclosingbox*

Specifies how the enclosing box is sized - based on the largest ligand, the displayed ligand or ligand length

Valid values: fitlargest

fitdisplayed

fitlength

Default value: **fitdisplayed**

*maxliglength*

The maximum ligand length expected.

Valid values: reals

Default value: **20**

Minimum: 0.0000000001

*recep\_ccut* The partial atomic charge below which receptor atoms are considered to be non-polar and will have their VDW radii scaled.

Valid values: reals

Default value: **0.25**

Minimum: 0.00000001

*recep\_vsclae*

The scaling factor for the VDW radii of non-polar receptor atoms.

Valid values:    reals  
Default value: **1**  
Minimum:        0.00000001

*xcent*              The X-coordinate of the enclosing box.

Valid values:    reals  
Default value: **1e-07**

*ycent*              The Y-coordinate of the enclosing box.

Valid values:    reals  
Default value: **1e-07**

*zcent*              The Z-coordinate of the enclosing box.

Valid values:    reals  
Default value: **1e-07**

## **grow**

Grows the currently selected fragment onto the currently selected grow bond.

Syntax:

## **grow**

## **growbond**

Specifies a grow bond for growing. A subsequent grow command will grow the currently selected fragment onto the this grow bond.

Syntax:

**growbond** <from\_atom> <to\_atom>

Operands:

<from\_atom> <to\_atom>

The two atoms which define the grow bond. In the growing procedure the “to” atom will be replaced by a suitable atom from the incoming fragment.

## **growbond2**

Specifies a second grow bond for growing. A subsequent grow command will grow the currently selected fragment onto the this grow bond.

Syntax:

**growbond2** <from\_atom> <to\_atom>

Operands:

<from\_atom> <to\_atom>

The two atoms which define the grow bond. In the growing procedure the “to” atom will be replaced by a suitable atom from the incoming fragment.

## **growdirection**

Sets the grow direction in the current fragment mode. The direction name must be one of the valid directions for the current fragment mode.

Syntax:

**growdirection** <direction\_name>

Operands:

<direction\_name>

The name of the grow direction which is to be made current. This must be a valid grow direction name within the current fragment.

## **growname**

Set the growname to that specified for all atoms which match the ASL specification.

Syntax:

**growname** <grow\_name> <ASL>

Operands:

<grow\_name> <ASL>

The first operand is the grow name which is to be applied to the atom. Only the first four characters of the growname will be used. The second operand is the ASL specification for all the atoms which are to have the growname applied.

## happly

This is a standard alias for **hydrogenapply** (see [hydrogenapply], page 194).

## hbondcriteria

Specify the criteria for calculating H-bonds in the workspace. The acceptor atom (A), H-bonded to the donor Hydrogen (H), is Oxygen, Nitrogen, Sulfur, or Fluorine.

Syntax:

**hbondcriteria** *acceptorangle*=⟨x⟩ *distance*=⟨x⟩ *donorangle*=⟨x⟩

Options:

*acceptorangle*

Minimum H — A-R angle, in degrees.

Valid values:   reals

Default value:   **90**

Minimum:       0.0

Maximum:       180.0

*distance*

Maximum separation between donor hydrogen atom and acceptor atom (H — A distance, in Angstroms).

Valid values:   reals

Default value:   **2.5**

Minimum:       0.0

*donorangle*

Minimum D-H — A angle, in degrees.

Valid values:   reals

Default value:   **120**

Minimum:       0.0

Maximum:       180.0

## **hbondset1**

Specify the first set of atoms used in finding H-bonds in the 3D workspace.

Syntax:

**hbondset1** <ASL>

Operands:

<ASL>

A string in the atom specification language. Typical usage is to define hbondset1 and hbondset2. This set, hbondset1, defines the “from” atoms. The hbondset2 atoms define the “to” atoms. H-bonds are calculated between these two sets. That is, the H-bonds are inter-set H-bonds. No intra-set H-bonds are calculated. If hbondset2’s ASL string is empty, then H-bonds are calculated for all atoms in hbondset1.

## **hbondset2**

Specify the second set of atoms used in finding H-bonds in the 3D workspace.

Syntax:

**hbondset2** <ASL>

Operands:

<ASL>

A string in the atom specification language. Typical usage is to define hbondset1 and hbondset2. This set, hbondset2, defines the “to” atoms. The hbondset1 atoms define the “from” atoms. H-bonds are calculated between these two sets. That is, the H-bonds are inter-set H-bonds. No intra-set H-bonds are calculated. If hbondset2’s ASL string is empty, then H-bonds are calculated for all atoms in hbondset1.

## **help**

This is a standard alias for **helpsearch** (see [helpsearch], page 181).

## **helpauto**

Turns on or off the automatic popup help.

Syntax:

**helpauto** *delay=⟨n⟩ on|off*

Options:

*delay*      The delay (in seconds) before the automatic help is popped up after the mouse enters the widget.

Valid values:    integers

Default value:    **2**

Operands:

*on|off*

If the operand is “on” then the popup help will be turned on. If off then it will be turned off.

## **helpcategory**

Changes the current help category

Syntax:

**helpcategory** ⟨category\_name⟩

Operands:

⟨category\_name⟩

The name of the category for which the help topics are to be displayed.

## **helpsearch**

Searches the help file for the specified search string. The search can be performed on all text or just the titles.

Syntax:

**helpsearch** *casesensitive=yes | no regularexpression=yes | no titleonly=yes | no <search\_string>*

Options:

*casesensitive*

A boolean option which determines if the search is performed in a case sensitive manner

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **false**

*regularexpression*

A boolean option which determines if the search string is to be treated as a regular expression

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **false**

*titleonly*

A boolean option which determines if the search applies to the title of the help topics or the entire text

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **false**

Operands:

*<search\_string>*

The text which the help file is to be searched for.

Aliases:

**help** (see [help], page 180)

## helptopic

Loads the topic given by the operand

Syntax:

**helptopic** *<topic\_name>*

Operands:

*<topic\_name>*

The name of the help topic to be displayed in the help panel. This must match the name of a topic exactly.

## **hideentries**

Hide the selected entry in the project table and create a subset of unselected entries. This function also switches to the subset view.

Syntax:

### **hideentries**

## **hidepanel**

Hide the panel whose name is given by the operands.

Syntax:

### **hidepanel <panel\_name>**

Operands:

<panel\_name>

The name of the panel which is to be hidden. The names available for use in the “**hidepanel**” command are displayed in parentheses after each item in the main menu bar.

## **hidepanels**

Hide all visible panels

Syntax:

### **hidepanels**

## **hideproperty**

This command hides the given property by creating a subset of all remaining properties in show state. This function also switch to the property subset view.

Syntax:

**hideproperty** <propertynname>

Operands:

<propertynname>

The name of the property to hide.

## **historyvisible**

Determines which types of commands will be visible in the command history list. Issuing a “historyvisible” alone will cause the list to be rebuilt with the current settings. A historyvisible all will make all commands in the list visible. A historyvisible readwrite=n will make all the read,write,sread and swrite commands hidden. Note that this only affects the visibility of the commands in the command history list of the script editor, not which commands are logged to a file.

Syntax:

```
historyvisible 1dplot=yes | no 2dplot=yes | no adjust=yes | no
alias=yes | no application=yes | no atomprop=yes | no
beginundoblock=yes | no bondprop=yes | no build=yes | no
clip=yes | no coloratom=yes | no debug=yes | no
delete=yes | no displayatom=yes | no displayopt=yes | no
entryexport=yes | no entryimport=yes | no
entrywscreate=yes | no errorcheck=yes | no fview=yes | no
find=yes | no fit=yes | no glide=yes | no helpauto=yes | no
helpcategory=yes | no helpsearch=yes | no helptopic=yes | no
historyvisible=yes | no hold=yes | no htreat=yes | no
impact=yes | no labelatom=yes | no liaison=yes | no
ligrep=yes | no macromodel=yes | no measurements=yes | no
monitor=yes | no pause=yes | no pausecommands=yes | no
pose=yes | no prefer=yes | no print=yes | no
project=yes | no qikprop=yes | no qsite=yes | no
quit=yes | no rename=yes | no rep=yes | no
ribbons=yes | no rotate=yes | no saveimage=yes | no
savelayout=yes | no saverestorereview=yes | no script=yes | no
sets=yes | no showhide=yes | no spotcenter=yes | no
superimpose=yes | no system=yes | no table=yes | no
tile=yes | no transformation=yes | no translate=yes | no
undoredo=yes | no zoom=yes | no [all]
```

Options:

*1dplot*

Valid values: boolean (true|false; yes|no; y|n; on|off)  
 Default value: **true**

*2dplot*

Valid values: boolean (true|false; yes|no; y|n; on|off)  
 Default value: **true**

*adjust*

Valid values: boolean (true|false; yes|no; y|n; on|off)  
 Default value: **true**

*alias*

Valid values: boolean (true|false; yes|no; y|n; on|off)  
 Default value: **true**

*application*

Valid values: boolean (true|false; yes|no; y|n; on|off)  
 Default value: **true**

*atomprop*

Valid values: boolean (true|false; yes|no; y|n; on|off)  
 Default value: **true**

<i>beginundoblock</i>	Valid values: Default value:	boolean (true false; yes no; y n; on off) <b>true</b>
<i>bondprop</i>	Valid values: Default value:	boolean (true false; yes no; y n; on off) <b>true</b>
<i>build</i>	Valid values: Default value:	boolean (true false; yes no; y n; on off) <b>true</b>
<i>clip</i>	Valid values: Default value:	boolean (true false; yes no; y n; on off) <b>true</b>
<i>coloratom</i>	Valid values: Default value:	boolean (true false; yes no; y n; on off) <b>true</b>
<i>debug</i>	Valid values: Default value:	boolean (true false; yes no; y n; on off) <b>true</b>
<i>delete</i>	Valid values: Default value:	boolean (true false; yes no; y n; on off) <b>true</b>
<i>displayatom</i>	Valid values: Default value:	boolean (true false; yes no; y n; on off) <b>true</b>
<i>displayopt</i>	Valid values: Default value:	boolean (true false; yes no; y n; on off) <b>true</b>
<i>entryexport</i>	Valid values: Default value:	boolean (true false; yes no; y n; on off) <b>true</b>
<i>entryimport</i>	Valid values: Default value:	boolean (true false; yes no; y n; on off) <b>true</b>
<i>entrywscreate</i>	Valid values: Default value:	boolean (true false; yes no; y n; on off) <b>true</b>
<i>errorcheck</i>	Valid values: Default value:	boolean (true false; yes no; y n; on off) <b>true</b>

<i>ffview</i>	Valid values:	boolean (true false; yes no; y n; on off)
	Default value:	<b>true</b>
<i>find</i>	Valid values:	boolean (true false; yes no; y n; on off)
	Default value:	<b>true</b>
<i>fit</i>	Valid values:	boolean (true false; yes no; y n; on off)
	Default value:	<b>true</b>
<i>glide</i>	Valid values:	boolean (true false; yes no; y n; on off)
	Default value:	<b>true</b>
<i>helpauto</i>	Valid values:	boolean (true false; yes no; y n; on off)
	Default value:	<b>true</b>
<i>helpcategory</i>	Valid values:	boolean (true false; yes no; y n; on off)
	Default value:	<b>true</b>
<i>helpsearch</i>	Valid values:	boolean (true false; yes no; y n; on off)
	Default value:	<b>true</b>
<i>helptopic</i>	Valid values:	boolean (true false; yes no; y n; on off)
	Default value:	<b>true</b>
<i>historyvisible</i>	Valid values:	boolean (true false; yes no; y n; on off)
	Default value:	<b>false</b>
<i>hold</i>	Valid values:	boolean (true false; yes no; y n; on off)
	Default value:	<b>true</b>
<i>htreat</i>	Valid values:	boolean (true false; yes no; y n; on off)
	Default value:	<b>true</b>
<i>impact</i>	Valid values:	boolean (true false; yes no; y n; on off)
	Default value:	<b>true</b>

*labelatom*

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **true**

*liaison*

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **true**

*ligrep*

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **true**

*macromodel*

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **true**

*measurements*

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **true**

*monitor*

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **true**

*pause*

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **true**

*pausecommands*

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **true**

*pose*

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **true**

*prefer*

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **true**

*print*

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **true**

*project*

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **true**

<i>qikprop</i>	Valid values:	boolean (true false; yes no; y n; on off)
	Default value:	<b>true</b>
<i>qsite</i>	Valid values:	boolean (true false; yes no; y n; on off)
	Default value:	<b>true</b>
<i>quit</i>	Valid values:	boolean (true false; yes no; y n; on off)
	Default value:	<b>true</b>
<i>rename</i>	Valid values:	boolean (true false; yes no; y n; on off)
	Default value:	<b>true</b>
<i>rep</i>	Valid values:	boolean (true false; yes no; y n; on off)
	Default value:	<b>true</b>
<i>ribbons</i>	Valid values:	boolean (true false; yes no; y n; on off)
	Default value:	<b>true</b>
<i>rotate</i>	Valid values:	boolean (true false; yes no; y n; on off)
	Default value:	<b>false</b>
<i>saveimage</i>	Valid values:	boolean (true false; yes no; y n; on off)
	Default value:	<b>true</b>
<i>savelayout</i>	Valid values:	boolean (true false; yes no; y n; on off)
	Default value:	<b>true</b>
<i>saverestoreview</i>	Valid values:	boolean (true false; yes no; y n; on off)
	Default value:	<b>true</b>
<i>script</i>	Valid values:	boolean (true false; yes no; y n; on off)
	Default value:	<b>true</b>
<i>sets</i>	Valid values:	boolean (true false; yes no; y n; on off)
	Default value:	<b>true</b>

*showhide*

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **true**

*spotcenter*

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **true**

*superimpose*

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **true**

*system*

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **true**

*table*

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **true**

*tile*

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **true**

*transformation*

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **false**

*translate*

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **false**

*undoredo*

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **true**

*zoom*

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **false**

Operands:

[all]

If “all” is specified then all commands will be visible in the command history list.

## hold

The hold command permits holding the current structures off screen.

Syntax:

**hold** <hold\_name>

Operands:

<hold\_name>

The name which will be applied to the hold set. This name will reference the hold in subsequent “addfromhold” and “replacefromhold” commands. If the name contains embedded spaces then it must be enclosed in quotation marks.

## hppmap

Sets the parameters for hppmap

Syntax:

**hppmap** *boxmargin*=<x> *boxtype*=box | ligand *cutoff*=<x>  
*cutoffscheme*=atom | neutral *gridspacing*=standard | high  
*incorporate*=append | replace | ignore *method*=h2o | dipole  
*source*=selected\_entries | workspace | file

Options:

*boxmargin* This value is used to increase the size of the box.

Valid values:   reals

Default value:   **6**

Minimum:       0.

*boxtype* This option indicates whether to treat the box ASL as defining a box or as defining a ligand.

Valid values:   box

                 ligand

Default value:   **box**

*cutoff* This specifies the cutoff value to use for the hppmap calculation, in angstroms.

Valid values:   reals

Default value:   **20**

Minimum:       0.

*cutoffscheme*

This option controls the cutoff scheme.

Valid values: atom  
neutral

Default value: **atom**

*gridspacing*

This option indicates what type of grid spacing (standard or high) to use for the hppmap.

Valid values: standard  
high

Default value: **standard**

*incorporate*

This option controls the incorporation of the results.

Valid values: append  
replace  
ignore

Default value: **replace**

*method* This option controls the method.

Valid values: h2o  
dipole

Default value: **dipole**

*source* This is the source for the structure.

Valid values: selected\_entries  
workspace  
file

Default value: **workspace**

## hppmapbox

Sets the box parameters for hppmap

Syntax:

**hppmapbox** < ASL >

Operands:

< ASL >

The ASL defines the box to use for the hppmap calculation.

## **hppmapset**

This command is used to specify the atoms for which the hppmap calculation will be performed.

Syntax:

**hppmapset** <ASL>

Operands:

<ASL>

The ASL defines the set of atoms for which the hppmap calculation is to be performed.

## **hppmapstart**

Starts a hppmap job with the current parameters.

Syntax:

**hppmapstart**

## **hppmapwrite**

Writes the hppmap input file with the current parameters.

Syntax:

**hppmapwrite** <file name>

Operands:

<file name>

Name of the file to write to. If the operand is blank, then use the default file name.

## **htreat**

This is a standard alias for **hydrogentreat** (see [hydrogentreat], page 194).

## **hydrogenapply**

Add or remove hydrogens and lone pairs to the atoms defined by the ASL operand according to the currently set treatment.

Syntax:

**hydrogenapply** <ASL>

Operands:

<ASL>

A string in the atom specification language. All atoms which match this specification will have hydrogens added or removed to become consistent with the current treatment.

Aliases:

**happly** (see [happly], page 179)

## **hydrogentreat**

Choose a treatment for hydrogen atom addition/deletion.

Syntax:

**hydrogentreat** <treatment\_name>

Operands:

<treatment\_name>

The operand is the name of a treatment which is to be made current. Note: treatment names are defined in \$SCHRODINGER/mmshare-vX.X/data/mmhtreat.ini. There are both long and short names for each treatments and either can be used.

Aliases:

**htreat** (see [htreat], page 193)

## **impactbufferedatom**

Specifies an atom to be buffered in an Impact calculation.

Syntax:

**impactbufferedatom** <atom\_number>

Operands:

<atom\_number>

The number of an atom which is to be treated as buffered during an Impact calculation.

## **impactbufferedset**

Specifies a set of atoms to be buffered in an Impact calculation.

Syntax:

**impactbufferedset** <ASL>

Operands:

<ASL>

The operand must be a valid string in the atom specification language.

## **impactconstraints**

Used to set all the options associated with constraints in impact

Syntax:

**impactconstraints** bonds=yes | no buffer\_force=<x>  
hmc\_bonds=yes | no hmc solvent=yes | no shake\_tolerance=<x>  
solvent=yes | no

Options:

*bonds* If true, then all bonds will be constrained during the Impact calculation.

Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **true**

*buffer\_force*

The buffered atom force constant

	Valid values:	reals
	Default value:	<b>25</b>
<i>hmcbonds</i>	If true, then all bonds will be constrained during the HMC calculation.	
	Valid values:	boolean (true false; yes no; y n; on off)
	Default value:	<b>false</b>
<i>hmcsolvent</i>	If true, then all solvent molecules will be held rigid during the HMC calculation.	
	Valid values:	boolean (true false; yes no; y n; on off)
	Default value:	<b>false</b>
<i>shake_tolerance</i>	The tolerance for the SHAKE (or RATTLE) algorithms.	
	Valid values:	reals
	Default value:	<b>1e-07</b>
	Minimum:	0.0
<i>solvent</i>	[NOTE: This option is no longer used.] If true, then all solvent molecules will be held rigid during the Impact calculation.	
	Valid values:	boolean (true false; yes no; y n; on off)
	Default value:	<b>true</b>

## **impactcontinuumsolvent**

Used to set all the options associated with the continuum solvent in Impact.

Syntax:

**impactcontinuumsolvent** *pfcutoff*=⟨x⟩ *resolution*=low | medium | high *sgbcutoff*=⟨x⟩ *type*=sgb | pbf | agbnp

Options:

<i>pfcutoff</i>	The type displacement threshold for the PBF calculation. <i>shake_tolerance</i> =d
	Valid values: reals Default value: <b>0.1</b> Minimum: 0.0

*resolution* The resolution for the PBF solvation calculation.

	Valid values:	low medium high
	Default value:	<b>low</b>
<i>sgbcutoff</i>	The type displacement threshold for the SGB calculation.	
	Valid values:	reals
	Default value:	<b>0.1</b>
	Minimum:	0.0
<i>type</i>	The type of continuum solvent which is to be used in the calculation.	
	Valid values:	sgb pbf agbnp
	Default value:	<b>sgb</b>

## impactdynamics

Settings associated with molecular dynamics simulations in Impact.

Syntax:

```
impactdynamics effectivedensity=⟨x⟩ ensemble=nvt | nve | npt
inittempgauss=⟨x⟩ initvelo=yes | no isothercomp=⟨x⟩
numbermdsteps=⟨n⟩ targetpress=⟨x⟩ targettemp=⟨x⟩
tautemp=⟨x⟩ tauvol=⟨x⟩ timestep=⟨x⟩
volumescaling=centerofmass | atom
```

Options:

### *effectivedensity*

The effective density used during an NPT Impact dynamics simulation.

Valid values:	reals
Default value:	<b>1</b>
Minimum:	0.00000000000001

*ensemble* The target ensemble to be achieved during an Impact dynamics simulation.

Valid values:	nvt nve npt
Default value:	<b>nvt</b>

*inittempgauss*

The temperature used to initialize the velocities from a Gaussian distribution during an NPT Impact dynamics simulation.

Valid values:   reals

Default value:   **298.15**

Minimum:       0.000000000001

*initvelo*

An option which determines if the velocities are to be initialized from a Gaussian distribution during an Impact dynamics simulation.

Valid values:   boolean (true|false; yes|no; y|n; on|off)

Default value:   **true**

*isothercomp*

The solvent isothermal compressibility to be used during an NPT Impact dynamics simulation.

Valid values:   reals

Default value:   **4.96e-05**

Minimum:       0.000000000001

*numbermdsteps*

The total number of time steps to be performed during an Impact dynamics simulation.

Valid values:   integers

Default value:   **100**

Minimum:       1

*targetpress*

The target pressure to be used during an NPT Impact dynamics simulation.

Valid values:   reals

Default value:   **1**

*targettemp*

The target temperature to be used during a NVT or NPT Impact dynamics simulation.

Valid values:   reals

Default value:   **298.15**

Minimum:       0.0000000001

*tautemp*

The temperature relaxation time to be used during a NVT or NPT Impact dynamics simulation.

Valid values:   reals

Default value:   **0.01**

Minimum:       0.0000000001

<i>tauvol</i>	The volume relaxation time to be used during an NPT Impact dynamics simulation.
Valid values:	reals
Default value:	<b>0.01</b>
Minimum:	0.0000000001
<i>timestep</i>	The time step to be used (in ps) during an Impact dynamics simulation.
Valid values:	reals
Default value:	<b>0.001</b>
Minimum:	0.0000001
<i>volumescaling</i>	The method of volume scaling to be used during an NPT Impact Dynamics simulation.
Valid values:	centerofmass atom
Default value:	<b>centerofmass</b>

## impactfastmultipole

Used to set all the options associated with the fast multipole method in Impact.

Syntax:

```
impactfastmultipole level=⟨n⟩ maximum=⟨n⟩
    smoothing=yes | no
```

Options:

<i>level</i>	The level for the fast multipole method.
Valid values:	integers
Default value:	<b>2</b>
Minimum:	1
<i>maximum</i>	The maximum for the fast multipole method.
Valid values:	integers
Default value:	<b>7</b>
Minimum:	4
Maximum:	20
<i>smoothing</i>	Whether or not to use smoothing with the fast multipole method.

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **true**

## **impactfrozenatom**

Specifies a single atom to be frozen at its current position in an Impact calculation.

Syntax:

**impactfrozenatom** <atom\_number>

Operands:

<atom\_number>

The number of an atom which is to be treated as frozen during an Impact calculation.

## **impactfrozensest**

Specifies a set of atoms to be frozen at their current positions in an Impact calculation.

Syntax:

**impactfrozensest** <ASL>

Operands:

<ASL>

A string in the atom specification language which describes the set of atoms which are to be treated as frozen in an Impact calculation.

## **impacthybridmc**

Settings associated with hybrid Monte Carlo simulations in Impact.

Syntax:

**impacthybridmc** *ncycles*=⟨n⟩ *nmdmc*=⟨n⟩ *timestep*=⟨x⟩

Options:

*ncycles* The total number of HMC cycles to be performed during an Impact hybrid Monte Carlo simulation.  
 Valid values: integers  
 Default value: **100**  
 Minimum: 1

*nmdmc* The number of MD steps per HMC cycle to be performed during an Impact hybrid Monte Carlo simulation.  
 Valid values: integers  
 Default value: **4**  
 Minimum: 1

*timestep* The time step to be used (in ps) during an Impact hybrid Monte Carlo simulation  
 Valid values: reals  
 Default value: **0.001**  
 Minimum: 0.0000001

## impactjob

This keyword is used to set various options associated with running Impact jobs.

Syntax:

**impactjob** *host*=⟨text⟩ *incorporate*=append | replace | ignore  
*job*=⟨text⟩ *login*=⟨text⟩ *structure\_source*=selected\_entries | workspace | file

Options:

*host* The name of the host for the Impact job.  
 Valid values: text strings  
 Default value:

*incorporate* How the results are to be incorporated into the project. This can be done with replacement of the existing entries, by appending as new entries to the project or by ignoring the final results.

	Valid values:      append replace ignore Default value: <b>append</b>
<i>job</i>	The name for the Impact job. Valid values: text strings Default value: <b>impacttmp</b>
<i>login</i>	The login name under which a Impact will be run. Valid values: text strings Default value:
<i>structure_source</i>	Whether to use the selected entries in the current project or what is in the workspace as input for the job. Valid values: selected_entries workspace file Default value: <b>workspace</b>

## impactmdparams

Used to set parameters associated with molecular dynamics simulations in Impact.

Syntax:

```
impactmdparams every=<n> fastfreq=<n> hmcintegrator=verlet
| rrespa integrator=verlet | rrespa mdstatistics=yes | no
mediumfreq=<n> nprint=<n> slowfreq=<n> stoprot=yes | no
traj=yes | no trajfile=<text> trajvelocities=yes | no
```

Options:

<i>every</i>	An option which determines how many MD steps between saving a frame of the trajectory for an Impact dynamics simulation. Valid values: integers Default value: <b>5</b>
<i>fastfreq</i>	RRESPA frequency for the fast forces. Valid values: integers Default value: <b>4</b> Minimum: <b>1</b>

*hmcintegrator*

Which integration technique is to be used during the Hybrid Monte Carlo simulation

Valid values:    verlet  
                  rrespa

Default value: **rrespa**

*integrator*

Which integration technique is to be used during the dynamics simulation

Valid values:    verlet  
                  rrespa

Default value: **verlet**

*mdstatistics*

An option which determines if statistics are to be gathered during an Impact dynamics simulation.

Valid values:    boolean (true|false; yes|no; y|n; on|off)

Default value: **false**

*mediumfreq*

RRESPA frequency for the medium forces.

Valid values:    integers

Default value: **2**

Minimum:        1

*nprint*

The number of steps at which output will be written during an Impact dynamics simulation.

Valid values:    integers

Default value: **5**

Minimum:        0

*slowfreq*

RRESPA frequency for the slow forces.

Valid values:    integers

Default value: **1**

Minimum:        1

*stoprot*

An option which determines if overall rotation and translation is stopped during the dynamics simulation.

Valid values:    boolean (true|false; yes|no; y|n; on|off)

Default value: **true**

*traj*

An option which determines if the trajectory is to be saved during an Impact dynamics simulation.

Valid values:    boolean (true|false; yes|no; y|n; on|off)

Default value: **false**

*trajfile*

The name of the trajectory file.

Valid values: text strings  
Default value: **trajectory.trj**

*trajvelocities*

An option which determines if the velocities are to be saved during an Impact dynamics simulation.

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **false**

## impactminimization

Used to set up a minimization in Impact.

Syntax:

```
impactminimization algorithm=newton | conjugate | steepest
    convergence=eadng | energy | gradient cutoff_1=<x>
    cutoff_2=<x> energy_change=<x> gradient=<x>
    initial_step_size=<x> maximum_step_size=<x> ncycles=<n>
    nfull_1=<n> nfull_2=<n> qsite_energychange=<x>
    qsite_gradient=<x> qsite_ncycles=<n>
    qsite_minalgorithm=newton | conjugate | steepest
```

Options:

*algorithm* Which algorithm is to be used for the minimization

Valid values: newton  
conjugate  
steepest  
Default value: **newton**

*convergence*

How convergence is to be established during the minimization.

Valid values: eadng  
energy  
gradient  
Default value: **eadng**

*cutoff\_1* Minimization parameter for Truncated Newton. Long range force cutoff.

Valid values: reals  
Default value: **10**  
Minimum: 0.0

*cutoff\_2* Minimization parameter for Truncated Newton. Long range force cutoff.

Valid values: reals

Default value: **10**

Minimum: 0.0

*energy\_change*

The energy change convergence criterion.

Valid values: reals

Default value: **1e-07**

Minimum: 0.0

*gradient* The gradient convergence criterion.

Valid values: reals

Default value: **0.01**

Minimum: 0.0

*initial\_step\_size*

The initial steps size for the minimization

Valid values: reals

Default value: **0.05**

Minimum: 0.0

*maximum\_step\_size*

The maximum step size for the minimization.

Valid values: reals

Default value: **1**

Minimum: 0.0

*ncycles*

The maximum number of minimization cycles to be performed in non-QSite jobs.

Valid values: integers

Default value: **100**

*nfull\_1*

Minimization parameter for Truncated Newton. Update long range forces every X steps.

Valid values: integers

Default value: **10**

Minimum: 0

*nfull\_2*

Minimization parameter for Truncated Newton. Update long range forces every X steps.

Valid values: integers

Default value: **10**

Minimum: 0

*qsite\_energychange*

The energy change convergence criterion for QSite.

Valid values:    reals  
Default value: **0.1**  
Minimum:        0.0

*qsite\_gradient*

The gradient convergence criterion for QSite.

Valid values:    reals  
Default value: **0.01**  
Minimum:        0.0

*qsite\_ncycles*

The maximum number of minimization cycles to be performed for QSite jobs.

Valid values:    integers  
Default value: **1000**

*qsiteminialgorithm*

Which algorithm is to be used for the minimization for qsite jobs.

Valid values:    newton  
                  conjugate  
                  steepest  
Default value: **conjugate**

## impactperiodicboundary

Used to set all the options associated with the periodic boundary conditions in Impact.

Syntax:

**impactperiodicboundary** *alpha=⟨x⟩ ewald=yes | no*  
                          *kvectormax=⟨n⟩ xsizes=⟨x⟩ ysize=⟨x⟩ zsize=⟨x⟩*

Options:

*alpha*      The alpha factor for the Ewald long-range correction.  
                  Valid values:    reals  
                  Default value: **0.25**  
                  Minimum:        0.0

*ewald*      Whether or not to use the Ewald long-range correction with the fast multipole method.  
                  Valid values:    boolean (true|false; yes|no; y|n; on|off)  
                  Default value: **false**

*kvectormax*

The maximum length of the k-space vectors in the Ewald long-range correction.

Valid values: integers

Default value: **5**

Minimum: 1

*xsize*

The size of the box in the X-dimension.

Valid values: reals

Default value: **18.62**

Minimum: 18.62

*ysize*

The size of the box in the Y-dimension.

Valid values: reals

Default value: **18.62**

Minimum: 18.62

*zsize*

The size of the box in the Z-dimension.

Valid values: reals

Default value: **18.62**

Minimum: 18.62

## **impactpotential**

Set various options associated with the definition of the potential energy to be used in a Impact job.

Syntax:

```
impactpotential continuum_solvent=yes | no dielectric=<x>
electrostatics=constant | distance_dependant
fast_multipole=yes | no field=oplsaa | opls1999 | opls2001 |
opls2005 paramfile=<text> periodic_boundary=yes | no
qsitefield=oplsaa | opls1999 | opls2001 | opls2005
truncate=yes | no
```

Options:

*continuum\_solvent*

Whether or not to use the continuum solvent

Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **false**

*dielectric*    The dielectric constant to be used in the electrostatic part of the energy calculation.

Valid values:    reals

Default value:    **1**

Minimum:    0.999999999

*electrostatics*

The electrostatic treatment to be used in the Impact calculation.

Valid values:    constant

                  distance-dependant

Default value:    **constant**

*fast\_multipole*

Whether or not to use the fast multipole method for electrostatic interactions.

Valid values:    boolean (true|false; yes|no; y|n; on|off)

Default value:    **false**

*field*

The force field to be used for the Impact calculation.

Valid values:    oplsaa

                  opls1999

                  opls2001

                  opls2005

Default value:    **opls2005**

*paramfile*

The name of the parameter file.

Valid values:    text strings

Default value:    **paramstd.dat**

*periodic\_boundary*

Whether or not to use the periodic boundary conditions

Valid values:    boolean (true|false; yes|no; y|n; on|off)

Default value:    **false**

*qsitefield*

The force field to be used for the Impact calculation.

Valid values:    oplsaa

                  opls1999

                  opls2001

                  opls2005

Default value:    **opls2001**

*truncate*

Whether or not to truncate the non-bonded interactions.

Valid values:    boolean (true|false; yes|no; y|n; on|off)

Default value:    **true**

## **impactstart**

Start an Impact input file with the current settings.

Syntax:

## **impactstart**

## **impacttask**

Determines which impact task is currently being set up.

Syntax:

## **impacttask** soak|mini|dynamics|hmc|qsite|glide|liaison

Operands:

soak|mini|dynamics|hmc|qsite|glide|liaison

The type of impact task which is to be set up. The operand cannot be abbreviated and must be given in full.

## **impacttruncation**

Used to set all the options associated with the truncation of non-bonded interactions in Impact.

Syntax:

## **impacttruncation** *distance*=⟨x⟩ *updatefrequency*=⟨n⟩

Options:

*distance*      The truncation distance for residue-based cutoffs.

Valid values:      reals

Default value:      **12**

Minimum:      0.0

*updatefrequency*

    The number of steps between update of the neighbor list.

    Valid values:      integers

Default value: **10**  
Minimum: 1

## **impactwrite**

Write an Impact input file with the current settings.

Syntax:

## **impactwrite**

## **invert**

Inverts the chirality around a chiral atom.

Syntax:

**invert** <chiral\_atom> <non\_moving\_atom1> <non\_moving\_atom2>

Operands:

<chiral\_atom> <non\_moving\_atom1> <non\_moving\_atom2>

Three atom numbers. The first is the atom around which the chirality is to be inverted. The second and third are atoms which are attached to the chiral atom but are not to be moved in the inversion process.

## **invertset**

Inverts all chiral centers in the specified set of atoms.

Syntax:

**invertset** <ASL>

Operands:

<ASL>

An ASL specification of the atoms which are to have their chiral centers inverted. Because of the way this works, this set of atom should involve at least whole molecules.

## **jaguarassignatomnames**

Sets the atom names of the matching atoms as per the jaguar standards i.e, element name + atom number. Incase of any duplicates, integer part of the name is incremented till it is unique in the Workspace

Syntax:

**jaguarassignatomnames** <ASL>

Operands:

<ASL>

The ASL expression describing the set of atoms which are to have their atom names made unique.

## **jaguarimportgeometry**

Import structures into the current project from a Jaguar input file.

Syntax:

**jaguarimportgeometry** *all=yes | no end=yes | no start=<n> total=<n> wsinclude=none | first | all wsreplace=yes | no [<filename>]*

Options:

*all* This determines whether all structures will be imported, or just a specified range.

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **true**

*end* This determines if all structures in the file are to be imported starting from the structure specified by start.

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **false**

<i>start</i>	This option sets the number of the first structure to be imported, if not importing all.  Valid values: integers Default value: <b>1</b> Minimum: 1
<i>total</i>	The total number of structures to be imported from the file, if not importing all structures.  Valid values: integers Default value: <b>1</b> Minimum: 1
<i>wsinclude</i>	This option determines which of the imported structures are to be included in the workspace. Valid values are “none”, “first”, or “all”.  Valid values: none first all Default value: <b>first</b>
<i>wsreplace</i>	This determines whether the structures currently in the workspace will be replaced by the included imported structures, or whether they will be included in the workspace also.  Valid values: boolean (true false; yes no; y n; on off) Default value: <b>true</b>

Operands:

[( filename )]

The name of the file from which structures will be imported. If no name is specified, then no import will be done.

## **jaguarinputfilesjob**

This keyword is used to set various options associated with running Jaguar input files jobs.

Syntax:

**jaguarinputfilesjob** *input\_files=( text )*

Options:

*input\_files*   Specifies existing structure file to be used as input files for running the job  
Valid values:    text strings  
Default value:

## **jaguarjob**

This keyword is used to set various options associated with running Jaguar jobs.

Syntax:

**jaguarjob** *structure\_files*=⟨text⟩ *structure\_source*=selected\_entries | workspace | file

Options:

*structure\_files*

If *structure\_source* is Selected structure files , specifies existing structure file to be used as input files for running the job.

Valid values:    text strings

Default value:

*structure\_source*

Whether to use the selected entries in the current project or what is in the workspace as input for the job.

Valid values:    selected\_entries  
                  workspace  
                  file

Default value:    **workspace**

## **jaguarselectextendtablerow**

Extends the selection to this row in the given orbital table in the Surfaces Property for Jaguar.

Syntax:

### **jaguarselectextendtablerow** *table=alpha | beta <row>*

Options:

<i>table</i>	Indicates whether to select a row in the alpha or beta orbital table.
Valid values:	alpha beta
Default value:	<b>alpha</b>

Operands:

*<row>*

The row number to extend the select to.

### **jaguarselectonlytablerow**

Selects only this row in the given orbital table in the Surface Property for Jaguar.

Syntax:

### **jaguarselectonlytablerow** *table=alpha | beta <row>*

Options:

<i>table</i>	Indicates whether to select a row in the alpha or beta orbital table.
Valid values:	alpha beta
Default value:	<b>alpha</b>

Operands:

*<row>*

The row number to select only in the table row.

### **jobcleanup**

Specifies a job to cleanup.

Syntax:

**jobcleanup** *files*=jobandmonitor | all <job\_id>

Options:

*files* This option determines which files are removed when the job is cleaned up. Either just the job record and monitoring files or all files associated with the job.

Valid values: jobandmonitor

all

Default value: **jobandmonitor**

Operands:

<job\_id>

The ID of the job which is to be cleaned up.

## jobsettings

This keyword is used to set various options associated with running a back-end job.

Syntax:

**jobsettings** *directory*=<text> *distributesubjobs*=maxprocessors | specifiedprocessors *host*=<text> *incorporate*=appendentries | replaceentries | ignoreentries | ingoreentries *jobname*=<text> *login*=<text> *maxstructures*=<n> *numcpus*=<n> *numsubjobs*=<n> *subjobsprocessors*=<n> *tmpdir*=<text> <model\_name>

Options:

*directory* The directory to write the output to.

Valid values: text strings

Default value:

*distributesubjobs*

How the subjobs are to be distributed over the available processors

Valid values: maxprocessors

specifiedprocessors

Default value: **maxprocessors**

*host*      The name of the host for the backend job.

Valid values:    text strings

Default value:    **host**

*incorporate*

How the results are to be incorporated into the project. This can be done by appending as new entries to the project or by ignoring the final results.

Valid values:    appendentries

replaceentries

ignoreentries

ingoreentries

Default value:    **appendentries**

*jobname*      The name for the backend job.

Valid values:    text strings

Default value:

*login*

The login name under which the job will be run.

Valid values:    text strings

Default value:    **user**

*maxstructures*

The maximum number of structures to be incorporated into the project.

Valid values:    integers

Default value:    **100**

Minimum:        0

*numcpus*      The number of CPUs to be used for jobs that support multiple CPUs.

Valid values:    integers

Default value:    **1**

Minimum:        1

*numsubjobs*

The number of subjobs this current job is to be run as.

Valid values:    integers

Default value:    **1**

Minimum:        1

*subjobsprocessors*

The number of processors to distribute the subjobs over

Valid values:    integers

Default value:    **1**

Minimum:        1

*tmpdir*      The path name of the scratch directory.

Valid values:    text strings

Default value:

Operands:

*{model\_name}*

The name of a model which has job settings like host, login, and so on.

## kill

This is a standard alias for **energykill** (see [energykill], page 100).

## labelatom

Displays labels for the atoms specified by the ASL operand.

Syntax:

```
labelatom 1charge=yes | no 2charge=yes | no acolor=yes | no
anum=yes | no atomicnumber=yes | no atomname=yes | no
atype=yes | no chain=yes | no chirality=yes | no
cindex={n} color={text} dmsopka=yes | no
element=yes | no entryname=yes | no font={text}
formalcharge=yes | no growname=yes | no h2opka=yes | no
headings=yes | no inscode=yes | no molnum=yes | no
molnumentry=yes | no numentry=yes | no nummol=yes | no
oneletter=yes | no pdbbfactor=yes | no pdbname=yes | no
resname=yes | no resnum=yes | no separator={text}
showlabel=yes | no stereochemistry=yes | no title=yes | no
user=yes | no utext={text} xoffset={x} xyz=yes | no
yoffset={x} {ASL}}
```

Options:

*1charge*    A boolean option which determines if the charge 1 value will be included in the atom label

Valid values:    boolean (true|false; yes|no; y|n; on|off)

Default value:    **false**

*2charge*    A boolean option which determines if the charge 2 value will be included in the atom label

	Valid values: boolean (true false; yes no; y n; on off) Default value: <b>false</b>
<i>acolor</i>	A boolean option which determines if labels will be colored the same as the atoms they label  Valid values: boolean (true false; yes no; y n; on off) Default value: <b>true</b>
<i>anum</i>	A boolean option which determines if the atom numbers will be included in the label  Valid values: boolean (true false; yes no; y n; on off) Default value: <b>true</b>
<i>atomicnumber</i>	A boolean option which determines if the atomic number will be included in the atom label  Valid values: boolean (true false; yes no; y n; on off) Default value: <b>false</b>
<i>atomname</i>	A boolean option which determines if the atom name will be included in the atom label  Valid values: boolean (true false; yes no; y n; on off) Default value: <b>false</b>
<i>atype</i>	A boolean option which determines if the MacroModel atom types will be included in the label  Valid values: boolean (true false; yes no; y n; on off) Default value: <b>false</b>
<i>chain</i>	A boolean option which determines if the chain name will be included in the label  Valid values: boolean (true false; yes no; y n; on off) Default value: <b>false</b>
<i>chirality</i>	A boolean option which determines if the atom chirality (R or S) will be included in the label.  Valid values: boolean (true false; yes no; y n; on off) Default value: <b>false</b>
<i>cindex</i>	An integer which indicates color index which is to be used for the labels. This will be ignored unless the acolor option is off  Valid values: integers Default value: <b>2</b> Minimum: <b>1</b> Maximum: <b>256</b>
<i>color</i>	A string which is the color name for atom labels. This will be ignored unless the acolor option is off

	Valid values: text strings Default value:
<i>dmsopka</i>	A boolean option which determines if the pka value in DMSO solvent will be included in the atom label Valid values: boolean (true false; yes no; y n; on off) Default value: <b>false</b>
<i>element</i>	A boolean option which determines if the element symbol will be included in the atom label Valid values: boolean (true false; yes no; y n; on off) Default value: <b>true</b>
<i>entryname</i>	A boolean option which determines if the entry name will be included in the atom label Valid values: boolean (true false; yes no; y n; on off) Default value: <b>false</b>
<i>font</i>	A string which determines which font is to be used for the labels Valid values: text strings Default value: <b>Arial Bold Small</b>
<i>formalcharge</i>	A boolean option which determines if the formal charge will be included in the atom label Valid values: boolean (true false; yes no; y n; on off) Default value: <b>false</b>
<i>growname</i>	A boolean option which determines if the grow name will be included in the atom label Valid values: boolean (true false; yes no; y n; on off) Default value: <b>false</b>
<i>h2opka</i>	A boolean option which determines if the pka value in H2O solvent will be included in the atom label Valid values: boolean (true false; yes no; y n; on off) Default value: <b>false</b>
<i>headings</i>	A boolean option which determines if the label fields will have headers in the labels (anum=1, atype=C2 and so on). Valid values: boolean (true false; yes no; y n; on off) Default value: <b>false</b>
<i>inscode</i>	A boolean option which determines if the pdb residue insertion code will be included in the label string Valid values: boolean (true false; yes no; y n; on off) Default value: <b>false</b>

<i>molnum</i>	A boolean option which determines if the molecule number will be included in the label Valid values: boolean (true false; yes no; y n; on off) Default value: <b>false</b>
<i>molnumentry</i>	A boolean option which determines if the molecule number by entry will be included in the label Valid values: boolean (true false; yes no; y n; on off) Default value: <b>false</b>
<i>numentry</i>	A boolean option which determines if the atom number by entry will be included in the label Valid values: boolean (true false; yes no; y n; on off) Default value: <b>false</b>
<i>nummol</i>	A boolean option which determines if the atom number by molecule will be included in the label Valid values: boolean (true false; yes no; y n; on off) Default value: <b>false</b>
<i>oneletter</i>	A boolean option which determines whether single letter pdb residue name is displayed or three letters. Valid values: boolean (true false; yes no; y n; on off) Default value: <b>false</b>
<i>pdbbfactor</i>	A boolean option which determines if the pdb bfactor will be included in the label string Valid values: boolean (true false; yes no; y n; on off) Default value: <b>false</b>
<i>pdbname</i>	A boolean option which determines if the pdb atom name will be included in the label string Valid values: boolean (true false; yes no; y n; on off) Default value: <b>false</b>
<i>resname</i>	A boolean option which determines if the pdb residue name will be included in the label string Valid values: boolean (true false; yes no; y n; on off) Default value: <b>false</b>
<i>resnum</i>	A boolean option which determines if the residue number will be included in the label string Valid values: boolean (true false; yes no; y n; on off) Default value: <b>false</b>
<i>separator</i>	A string option which is the inserted between each field in the label string.

	Valid values:	text strings
	Default value:	
<i>showlabel</i>	A boolean option which determines if an atom label is displayed when the atom itself is undisplayed.	
	Valid values:	boolean (true false; yes no; y n; on off)
	Default value:	<b>false</b>
<i>stereochemistry</i>	A boolean option which determines if the double bond stereochemistry (E or Z) will be included in the label.	
	Valid values:	boolean (true false; yes no; y n; on off)
	Default value:	<b>false</b>
<i>title</i>	A boolean option which determines if the entry title will be included in the atom label	
	Valid values:	boolean (true false; yes no; y n; on off)
	Default value:	<b>false</b>
<i>user</i>	A boolean option which determines if the user-defined text will be included in the label	
	Valid values:	boolean (true false; yes no; y n; on off)
	Default value:	<b>false</b>
<i>utext</i>	This option sets the user defined text for atom labels	
	Valid values:	text strings
	Default value:	
<i>xoffset</i>	A double option which sets the x-offset from the atom center at which the labels are to appear.	
	Valid values:	reals
	Default value:	<b>0.05</b>
<i>xyz</i>	A boolean option which determines if the xyz location of the atom will be included in the label.	
	Valid values:	boolean (true false; yes no; y n; on off)
	Default value:	<b>false</b>
<i>yoffset</i>	A double option which sets the y-offset from the atom center at which the labels are to appear.	
	Valid values:	reals
	Default value:	<b>-0.15</b>

Operands:

*<ASL>*

A string in the atom specification language. All atoms which match this specification will be labeled with the current label settings.

## **labelclear**

Clears labels from the atoms specified by the ASL operand.

Syntax:

**labelclear** <ASL>

Operands:

<ASL>

A string in the atom specification language. All atoms which match this specification will have their labels removed.

## **labelformat**

Sets the label format string which will be used in subsequent labeling operations.

Syntax:

**labelformat** <format\_string>

Operands:

<format\_string>

A string which defines the current labeling format. See on-line help for details.

## **labelupdate**

Apply the current label format to all the atoms which are labeled.

Syntax:

**labelupdate** [NON\_EL]

Operands:

[NON\_EL]

A string which defines the relabeling method according to its caller (NON\_EL for relabeling only atom labels, excluding the existing element labels. Otherwise both elementlabels and atom labels are relabelled).

## **liaisonanalysis**

Used to do the analysis

Syntax:

```
liaisonanalysis liaisonalpha=<x> liaisonanalysistype=fit | predict
    liaisonbefile=<text> liaisonbeta=<x>
    liaisonenergymodeltype=equation | glidescore
    liaisonfitname=<text> liaisonfixalpha=yes | no
    liaisonfixbeta=yes | no liaisonfixgamma=yes | no
    liaisonfixintercept=yes | no liaisonfixslope=yes | no
    liaisongamma=<x> liaisonintercept=<x> liaisonlistfile=<text>
    liaisonslope=<x> liaisonspecligs=readligs | enterligs
```

Options:

### *liaisonalpha*

This option sets the van der Waals (Alpha) coefficient.

Valid values:    real

Default value:    **0**

### *liaisonanalysistype*

This option determines the analysis type, including two options of Predict (predict) and Fit (fit).

Valid values:    fit

                  predict

Default value:    **fit**

### *liaisonbefile*

This option determines the name of the ligand binding energy file

Valid values:    text strings

Default value:

### *liaisonbeta*

This option sets the electrostatic (Beta) coefficient.

Valid values:    real

Default value:    **0**

*liaisonenergymodeltype*

This option determines which binding energy model is used, LIA equation or Glidescore

Valid values:    equation  
                  glidescore

Default value: **equation**

*liaisonfitname*

This option determines the names of ligands to be predicted

Valid values:    text strings  
Default value:

*liaisonfixalpha*

Allows the alpha value to be used as a constraint.

Valid values:    boolean (true|false; yes|no; y|n; on|off)

Default value: **false**

*liaisonfixbeta*

Allows the beta value to be used as a constraint.

Valid values:    boolean (true|false; yes|no; y|n; on|off)

Default value: **false**

*liaisonfixgamma*

Allows the gamma value to be used as a constraint

Valid values:    boolean (true|false; yes|no; y|n; on|off)

Default value: **false**

*liaisonfixintercept*

Allows the Glidescore intercept to be used as a constraint

Valid values:    boolean (true|false; yes|no; y|n; on|off)

Default value: **false**

*liaisonfixslope*

Allows the glidescore slope to be used as a constraint.

Valid values:    boolean (true|false; yes|no; y|n; on|off)

Default value: **false**

*liaisongamma*

This option sets cavity (Gamma) coefficient.

Valid values:    reals

Default value: **0**

*liaisonintercept*

This option sets the Glidescore intercept

Valid values:    reals

Default value: **0**

*liaisonlistfile*

This option determines the name of file that contains names of ligands to be predicted

Valid values: text strings  
 Default value:

*liaisonslope*

This option sets the Glidescore slope

Valid values: reals  
 Default value: **0**

*liaisonspecligs*

This option determines the methods used to specify the ligands either by reading ligand names from a text file (readligs), or entering a comma separated list of ligand names (enterligs ).

Valid values: readligs  
 enterligs  
 Default value: **readligs**

**liaisonparameters**

Used to specify simulation parameters in a Liasion (Linear Interaction Approximation).

Syntax:

```
liaisonparameters algorithm=newton | conjugate | steepest
  liaisonboundgcut=<x> liaisonboundheat=<x>
  liaisonboundint=<n> liaisonboundmini=<n>
  liaisonboundpremini=<n> liaisonboundprod=<x>
  liaisonfreeint=<n> liaisonliggcut=<x> liaisonligheat=<x>
  liaisonligmini=<n> liaisonligpremini=<n> liaisonligprod=<x>
  liaisonmethod=mini | hmc | dyn liaisonreltime=<x>
  liaisonrescut=<x> liaisontemp=<x> minibound=yes | no
  minilig=yes | no
```

Options:

*algorithm* Which algorithm is to be used for the minimization

Valid values: newton  
 conjugate  
 steepest  
 Default value: **newton**

*liaisonboundgcut*

This option determines the RMS gradient for convergence (in kcal/mol/A)

Valid values:    reals  
Default value: **0.05**

*liaisonboundheat*

This option determines the time for heating/ equilibration (ps)

Valid values:    reals  
Default value: **5**  
Minimum:        0.0

*liaisonboundint*

This option determines the number of steps for collecting Liaison statistics

Valid values:    integers  
Default value: **10**  
Minimum:        0

*liaisonboundmini*

This option determines the maximum number of minimization steps

Valid values:    integers  
Default value: **500**  
Minimum:        0

*liaisonboundpremini*

This option determines the maximum number of minimization steps

Valid values:    integers  
Default value: **500**  
Minimum:        0

*liaisonboundprod*

This option determines the time for data collection (in ps)

Valid values:    reals  
Default value: **5**  
Minimum:        0.0

*liaisonfreeint*

This option determines the number of steps for collecting Liasion statistics

Valid values:    integers  
Default value: **10**  
Minimum:        0

*liaisonliggcut*

This option determines the RMS gradient for convergence

Valid values:   reals  
Default value: **0.01**  
Minimum:       0.0

*liaisonlighat*

This option determines the time for heating/ equilibration (ps)

Valid values:   reals  
Default value: **5**  
Minimum:       0.0

*liaisonligmini*

This option determines the maximum number of minimization steps

Valid values:   integers  
Default value: **500**  
Minimum:       0

*liaisonligpremini*

This option determines maximum number of minimization steps

Valid values:   integers  
Default value: **1000**  
Minimum:       0

*liaisonligprod*

This option determines the time for data collection (in ps)

Valid values:   reals  
Default value: **5**  
Minimum:       0.0

*liaisonmethod*

This option determines which sampling method will be used during a Liaison analysis. The 3 options are minimization (mini), Hybrid Monte Carlo (hmc) and Molecular dynamics (dyn).

Valid values:   mini  
                  hmc  
                  dyn  
Default value: **mini**

*liaisonreltime*

This option determines the temperature relaxation time

Valid values:   reals  
Default value: **0.01**  
Minimum:       0.0

*liaisonrescut*

This option determines the residue-based cutoff for nonbonded interactions

Valid values:   reals

Default value:   **15**

Minimum:       0.0

*liaisontemp*

This option determines the simulation temperature

Valid values:   reals

Default value:   **300**

Minimum:       0.0

*minibound*

This option determines whether the ligand-receptor complex is minimized before running simulation.

Valid values:   boolean (true|false; yes|no; y|n; on|off)

Default value:   **true**

*minilig*

This option determines whether the ligand is minimized before running the simulation

Valid values:   boolean (true|false; yes|no; y|n; on|off)

Default value:   **true**

## **liaisonselectlig**

Defines a on-screen molecule to be treated as the ligand for a Liaison calculation

Syntax:

**liaisonselectlig** <molecule\_number>

Operands:

<molecule\_number>

The number of a molecule to be included as the ligand.

## **liaisonsettings**

Used to set values associated with the Linear Interaction Approximation (Liaison)

Syntax:

**liaisonsettings** *liaisonsubdir*=⟨text⟩ *liaisontype*=sim | analyze  
*numproc*=⟨n⟩

Options:

*liaisonsubdir*

This option determines the subdirectory to use for Liaison jobs.

Valid values: text strings

Default value: **liaison**

*liaisontype*

This option determines the job type for Liaison analysis. There are two types: (1) Simulate (sim); (2) Analyze results of earlier simulations (analyze)

Valid values: sim  
analyze

Default value: **sim**

*numproc*

This option determines the number of processors the job is to be run on.

Valid values: integers

Default value: **1**

Minimum: 1

## liaisonsystem

Used to specify receptor and ligand(s) to be simulated in the Linear Interaction Approximation (Liaison)

Syntax:

**liaisonsystem** *haslialig*=yes | no *liaisonligfmt*=m2io | sd | mol2 | pdb *liaisonligfname*=⟨text⟩ *liaisonsimtype*=mult | single *liaisonsname*=⟨text⟩ *ligftype*=ligstruct | liglist *usepartialcharges*=yes | no

Options:

*haslialig*

This option determines whether the displayed structure includes the ligands

Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **true**

*liaisonligfmt*

This option for selecting the format of the ligand file. The four possible formats are MAESTRO (m2io), MDL SD (sd), MOL2 (mol2) and PDB (pdb)

Valid values: m2io  
sd  
mol2  
pdb

Default value: **m2io**

*liaisonligfname*

This option determines the name of the ligand file

Valid values: text strings  
Default value:

*liaisonsimtype*

This option determines the type of ligand(s)-receptor structures to be simulated, having two options: (1) Multiple ligands, single receptor; (2) Single ligand, single receptor.

Valid values: mult  
single  
Default value: **mult**

*liaisonsname*

This option determines the name to be used for this ligand

Valid values: text strings  
Default value: **Lig**

*ligtype*

This option determines the source of ligand structure(s), having two options: (1) File containing a single ligand structure ( ligstruct ); and (2) File containing a list of ligand names and structure files ( liglist ).

Valid values: ligstruct  
liglist  
Default value: **ligstruct**

*usepartialcharges*

This option determines whether to use partial charges from the Maesro file

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **false**

## **ligandbond**

Specifies two atoms which define a bond to be treated as a ligand bond during a conformational search.

Syntax:

**ligandbond** <atom1> <atom2>

Operands:

<atom1> <atom2>

The numbers of two atoms which are to be treated as a ligand bond during a conformational search. Note that specifying a-b is the same as specifying b-a.

## **lightambient**

Sets the ambient intensity of the light.

Syntax:

**lightambient** *intensity*=<x>

Options:

*intensity*    Ambient lighting component.

Valid values:    reals

Default value:    **0.1**

Minimum:    0.0

Maximum:    1.0

## **lightdiffuse**

Sets the diffuse intensity of the light.

Syntax:

**lightdiffuse** *intensity=(x)*

Options:

*intensity* Diffuse lighting component.

Valid values: reals

Default value: **1**

Minimum: 0.0

Maximum: 1.0

**lightposition**

Sets the position of the light.

Syntax:

**lightposition** *(x) (y) (z)*

Operands:

*(x) (y) (z)*

The position of the light.

**lightspecular**

Sets the specular intensity of the light.

Syntax:

**lightspecular** *intensity=(x)*

Options:

*intensity* Specular lighting component.

Valid values: reals

Default value: **1**

Minimum: 0.0

Maximum: 1.0

## ligprep

This keyword is used to set various options associated with running LigPrep jobs.

Syntax:

```
ligprep desalt=yes | no filter_file=<text> forcefield=mmffs |
           opls2005 gen_conform=<n> gen_stereo=<n>
           gen_tautomers=yes | no input_file=<text> ionization=generate
           | neutralize | retain ionizationmethod=ionizer | epik
           output_format=maestro | sdf ph=<x> ph_tolerance=<x>
           stereoisomers=retain | determine | generate
           structure_source=selected_entries | workspace | file
```

Options:

*desalt* Desalt  
 Valid values: boolean (true|false; yes|no; y|n; on|off)  
 Default value: **true**

*filter\_file* The name of the filter criteria file.  
 Valid values: text strings  
 Default value:

*forcefield* Force field to use for minimization and filtering  
 Valid values: mmffs  
 opls2005  
 Default value: **opls2005**

*gen\_conform* Generate low energy ring confirmations  
 Valid values: integers  
 Default value: **1**  
 Minimum: 1

*gen\_stereo* Generate stereoisomers (maximum)  
 Valid values: integers  
 Default value: **32**  
 Minimum: 1

*gen\_tautomers* Generate tautomers  
 Valid values: boolean (true|false; yes|no; y|n; on|off)  
 Default value: **true**

*input\_file* The name of the structure input file.

Valid values: text strings

Default value:

*ionization* How to generate the ionization states

Valid values: generate  
neutralize  
retain

Default value: **generate**

*ionizationmethod*

Which ionization computation method to use (default or Epik)

Valid values: ionizer

epik

Default value: **ionizer**

*output\_format*

Whether to output the file in Maestro or SDF format.

Valid values: maestro  
sdf

Default value: **maestro**

*ph* Ionization pH

Valid values: reals

Default value: **7**

Minimum: 0.0

Maximum: 14.0

*ph\_tolerance*

Ionization pH tolerance

Valid values: reals

Default value: **2**

Minimum: 0.0

Maximum: 7.0

*stereoisomers*

How to generate the stereoisomers

Valid values: retain  
determine  
generate

Default value: **retain**

*structure\_source*

Whether to use the selected entries in the current project, or what is in the workspace, or a specified file with multiple structures as structure input for the job.

Valid values: selected\_entries  
workspace  
file  
Default value: **file**

## **ligprepstart**

Start a LigPrep job with the current settings.

Syntax:

## **ligprepstart**

## **localatoms**

Specify the group of local (transformation) atoms

Syntax:

## **localatoms < ASL >**

Operands:

**< ASL >**

For specifying which atoms are to be locally transformed. A list of atoms is created at the time the command is issued. Use the transform command to specify local scope, then use the rotate and/or translate commands to perform local transformations. To get back to global scope use the transform command to set the scope back to global.

## **localbitset**

Specify whether local bitsets (atoms and center) are updated whenever contents of Workspace change. Unless the lock is enabled, these bitsets are re-evaluated each time atoms are changed, including changes in atom coordinates due to local adjustment.

Syntax:

**localbitset** *lock=yes | no*

Options:

*lock*      Enable/disable lock on local bitsets for as long as ASL expressions and number of atoms in Workspace remain unchanged.  
Valid values:    boolean (true|false; yes|no; y|n; on|off)  
Default value:   **false**

## **localcenter**

Set local transformation center to given centroid

Syntax:

**localcenter**

## **logfile**

This is a standard alias for **scriptlogfile** (see [scriptlogfile], page 435).

## **logp**

Support calculation of LogP of MacroModel.

Syntax:

**logp** *enable=yes | no secsolvent=none | water | chcl3 | octanol*

Options:

*enable*      Determines whether logp will be used with this multiple minimization.  
Valid values:    boolean (true|false; yes|no; y|n; on|off)  
Default value:   **false**

**secsolvent** This option determines the second solvent for logp. soft-limit=<n> <input\_file\_name>

Valid values: none  
water  
chcl3  
octanol

Default value: **water**

## **macrodefine**

Define a macro. Definition is a semicolon separated string of maestro commands.

Syntax:

**macrodefine** <macro\_name> <definition>

Operands:

<macro\_name> <definition>

The first operand is the name of the macro. If this contains embedded spaces then it must be enclosed in double quotes. The second operand is the definition. If this contains embedded spaces then it must be enclosed in double quotes. To run the macro, execute macrorun <macro\_name>

## **macrorun**

Invokes an already defined macro

Syntax:

**macrorun** <macro\_name>

Operands:

<macro\_name>

The operand is the name of the macro. If this contains embedded spaces then it must be enclosed in double quotes. As a shortcut macros can also be run without preceding them with macrorun . Using macrorun has the advantage that if the macro doesn't exist the program will tell you the macro doesn't exist. If you omit macrorun and Maestro cannot find the macro, then

Maestro will report that the command does not exist. This will be misleading and possibly confusing because it's really the macro that will not exist.

## **makedirectory**

Create new directory.

Syntax:

**makedirectory** <directory\_name>

Operands:

<directory\_name>

The name of the directory to created.

## **materialmolecular**

Syntax:

**materialmolecular** *shininess*=<x> *specular*=<x>

Options:

*shininess* Shininess component for material of molecules

Valid values: reals  
Default value: **0.5**  
Minimum: 0.0  
Maximum: 1.0

*specular* Specular component for material of molecules

Valid values: reals  
Default value: **0.7**  
Minimum: 0.0  
Maximum: 1.0

## **mcsd**

A command which defines settings for the MC/SD (Monte Carlo/Stochastic Dynamics ) simulation.

Syntax:

**mcsd** *maxdof=⟨ n ⟩ mindof=⟨ n ⟩ ratio=⟨ n ⟩ temperature=⟨ x ⟩*

Options:

*maxdof*      The maximum number of torsions which will will be varied at each MC trial.

Valid values:    integers

Default value:    **1**

Minimum:        1

*mindof*      The minimum number of torsions which will will be varied at each MC trial.

Valid values:    integers

Default value:    **1**

Minimum:        1

*ratio*           The ratio of Monte Carlo to Stochastic Dynamics steps in the simulation.

Valid values:    integers

Default value:    **1**

Minimum:        1

*temperature*

The temperature at which the MCSD simulation is to be performed. [NOTE: This option is no longer used.]

Valid values:    reals

Default value:    **300**

Minimum:        0.0

## **mini**

This is a standard alias for **minienergy** (see [[minienergy](#)], page 240).

## minienergy

Used to set values associated with a MacroModel energy minimization

Syntax:

```
minienergy converge=nothing | energy | gradient | movement  
    maxiter=<n> method=sd | prcg | osvm | fmnr | tncg | lbfgs |  
    optimal threshold=<x>
```

Options:

*converge* This option determines which convergence criterion will be used during an energy minimization.

Valid values: nothing  
energy  
gradient  
movement

Default value: **gradient**

*maxiter* This option determines the maximum number of iterations which will be performed during an energy minimization.

Valid values: integers  
Default value: **500**  
Minimum: 0  
Maximum: 9999999

*method* This option determines which minimization method will be used.

Valid values: sd  
prcg  
osvm  
fmnr  
tncg  
lbfgs  
optimal  
Default value: **prcg**

*threshold* This option determines what the convergence threshold will be.

Valid values: reals  
Default value: **0.05**  
Minimum: 0.0

Aliases:

**mini** (see [mini], page 239)

## **minta**

Used to set up options associated with BatchMin molecular energy calculation.

Syntax:

```
minta hardlimit=⟨x⟩ numenergy=⟨n⟩ numint=⟨n⟩ softlimit=⟨n⟩
          temperature=⟨x⟩ ⟨input_file_name⟩
```

Options:

*hardlimit* The hard limit for sampling along normal modes.

Valid values: reals

Default value: **1**

Minimum: 0.0

Maximum: 3.0

*numenergy*

This option determines the number of energy evaluations per MINTA integration.

Valid values: integers

Default value: **2000**

Minimum: 1

*numint*

This option determines the number of MINTA iteration.

Valid values: integers

Default value: **5**

Minimum: 1

*softlimit*

The soft limit for sampling along normal modes.

Valid values: integers

Default value: **3**

Minimum: 1

Maximum: 3

*temperature*

The temperature for MINTA calculation.

Valid values: reals

Default value: **300**

Minimum: 0.0

Operands:

⟨*input\_file\_name*⟩

The name of the input file. This name must be given in full, including any suffix.

## **monitor**

This is a standard alias for **energymonitor** (see [energymonitor], page 101).

## **monitorangle**

Specifies a triplet of atoms to have their bond angle monitored during a MacroModel dynamics simulation.

Syntax:

**monitorangle** <atom1> <atom2> <atom3>

Operands:

<atom1> <atom2> <atom3>

The atom numbers of three atoms which are to have the angle between them monitored during a dynamics simulation. Note that specifying a-b-c is the same as c-b-a

## **monitordistance**

Specifies a pair of atoms to have their distance monitored during a Macro-Model dynamics simulation.

Syntax:

**monitordistance** <atom1> <atom2>

Operands:

<atom1> <atom2>

The atom numbers of two atoms which are to have the distance between them monitored during a dynamics simulation. Note that specifying a-b is the same as b-a

## monitorhbond

Specifies a quartet of atoms which define an H-bond to be monitored during a dynamics simulation. The atoms are specified as X-H...Y-Z e.g. N2-H3...O2=C2 for a amide-amide bond.

Syntax:

```
monitorhbond distance=⟨x⟩ hyzangle=⟨x⟩ xhyangle=⟨x⟩  
⟨xatom⟩ ⟨hatom⟩ ⟨yatom⟩ ⟨zatom⟩
```

Options:

*distance*      Specifies the maximum H...Y distance for an acceptable hydrogen bond.

Valid values:      reals  
Default value:      **2.5**  
Minimum:      0.0

*hyzangle*      Specifies the minimum H...Y-Z angle for an acceptable hydrogen bond.

Valid values:      reals  
Default value:      **90**  
Minimum:      0.0  
Maximum:      180.0

*xhyangle*      Specifies the minimum X-H...Y angle for an acceptable hydrogen bond.

Valid values:      reals  
Default value:      **120**  
Minimum:      0.0  
Maximum:      180.0

Operands:

⟨xatom⟩ ⟨hatom⟩ ⟨yatom⟩ ⟨zatom⟩

The first operand is the atom number of the heavy atom (X) to which the donor hydrogen is attached. The second operand is the atom number of the donor hydrogen itself (H). The third operand is the atom number of acceptor atom (Y) and the fourth operand is the atom number of the heavy atom (Z) attached to the acceptor.

## **monitorsetsurf**

Specifies a set of atoms to have their surface areas monitored during a MacroModel molecular dynamics simulation.

Syntax:

**monitorsetsurf** *<ASL>*

Operands:

*<ASL>*

A string in the atom specification language which describes the set of atom which are to have their surface areas monitored during a dynamics simulation.

## **monitorsettings**

Controls settings for how the monitor panel in Maestro functions. At present it's just for whether jobs for the current project are displayed or all jobs.

Syntax:

**monitorsettings** *onlycurrentproj=yes | no*

Options:

*onlycurrentproj*

This determines whether the jobs listed in the monitor panel are limited to those for the current project or all known jobs for the current user.

Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **true**

## **monitorsurf**

Specifies a single atom to have its surface area monitored during a Macro-Model molecular dynamics simulation.

Syntax:

### **monitorsurf** <atom\_number>

Operands:

<atom\_number>

The number of an atom which is to have its surface areas monitored during a dynamics simulation.

### **monitortorsion**

Specifies a quartet of atoms to have their torsion angle monitored during a MacroModel dynamics simulation.

Syntax:

**monitortorsion** <atom1> <atom2> <atom3> <atom4>

Operands:

<atom1> <atom2> <atom3> <atom4>

The atom numbers of four atoms which are to have the torsion between them monitored during a dynamics simulation. Note that specifying a-b-c-d is the same as d-c-b-a

### **mouse**

Specify how mouse movements are to be interpreted

Syntax:

**mouse** *rotate=xy | x | y | z | adjust* *translate=xy | x | y | z*

There are no operands for this command.

Options:

*rotate* How mouse movements are interpreted for rotation. Default is rotate in xy.

Valid values:      xy  
                      x  
                      y  
                      z  
                      adjust

Default value: **xy**

*translate* How mouse movements are interpreted for translation. Default is translate in xy.

Valid values: **xy**

**x**

**y**

**z**

Default value: **xy**

Operands:

There are no operands for this command.

## **move**

Moves the specified atoms by the offsets specified in x, y and z values

Syntax:

**move** <atom\_num> <xinc> <yinc> <zinc>

Operands:

<atom\_num> <xinc> <yinc> <zinc>

The first operand is the number of the atom which is to be moved. The following three real numbers represent the offsets to be applied to the x, y and z coordinates in Angstroms.

## **multiplemini**

Used to set up a multiple minimization. The operand is the name of a file which contains a number of structures.

Syntax:

**multiplemini** *maxdist*=<x> *window*=<x> <input\_file\_name>

Options:

*maxdist* Maximum distance between atoms in equal structures.

Valid values: **reals**

Default value: **0.25**  
Minimum: 0.0

*window*      Specifies the energy window (in kJ/mol) for saving structures during a multiple minimization.

Valid values:    reals  
Default value: **50**  
Minimum: 0.0

Operands:

*<input\_file\_name>*

The name of the input file. This name will contain all the structures which are to be minimized. This name must be given in full, including any suffix.

## **mutate**

Replace the sidechain of any residue in set <ASL> with the sidechain of the currently selected fragment

Syntax:

**mutate** <ASL>

Operands:

*<ASL>*

The set of atoms which are to be mutated. Each residue represented in this set will be mutated.

## **nextpose**

If there is more than one pose structure in the pose viewer, display the one after the last displayed pose. Upon reaching the end of the list, wraps around to the beginning.

Syntax:

**nextpose**

**optimizefog**

Fit clipping planes to displayed structure to maximize the shading contrast between near and far atoms.

Syntax:

**optimizefog**

**partialcharge**

Set the partial atomic charges for all atoms which match the ASL specification.

Syntax:

**partialcharge** ⟨charge1⟩ ⟨charge2⟩ ⟨ASL⟩

Operands:

⟨charge1⟩ ⟨charge2⟩ ⟨ASL⟩

The first operand must be a valid real number which will be used for the charge1 field of all matching atoms. The second operand must be a valid real number which will be used for the charge2 field of all matching atoms. The final operand is a valid ASL string which specifies which atoms are to have their charges changed.

**pause**

Pauses immediately for the length of time specified (in seconds, unless other units are specified). Setting a pause of zero (e.g. “0” or “0.00”) will mean there is no pause. If no operand is specified, or a negative time is given, then Maestro will remain paused indefinitely, until it is interrupted. All pauses can be interrupted by pressing any key on the keyboard or any mouse button.

Syntax:

**pause** ⟨ pause\_time ⟩ [sec|min|hour|day]

Operands:

⟨ pause\_time ⟩ [sec|min|hour|day]

The length of time to pause - floating point value in seconds, accurate to within about 0.05 second.

## **pausecommands**

Sets the time to pause after execution of every command in script (in seconds, unless other units are specified). Setting a pause of zero (e.g. “0” or “0.00”) will mean there is no pause after each command. If no operand is specified, or a negative time is given, then the pause after each command will be indefinite, pausing until interrupted. All pauses can be interrupted by pressing any key on the keyboard or any mouse button.

Syntax:

**pausecommands** ⟨ pause\_time ⟩ [sec|min|hour|day]

Operands:

⟨ pause\_time ⟩ [sec|min|hour|day]

The time to pause after each command - floating point value in seconds, accurate to within about 0.05 second.

## **phaseaddcustomfeature**

Adds the given custom feature.

Syntax:

**phaseaddcustomfeature** *code*=⟨ text ⟩ ⟨ name ⟩

Options:

*code*      The name of the custom feature to add.

Valid values:    text strings

Default value:    **A**

Operands:

$\langle \text{name} \rangle$

The name for the feature.

## phaseaddligands

Adds the given ligands to the table.

Syntax:

**phaseaddligands** *activity*= $\langle \text{text} \rangle$  *convert\_activity*=yes | no  
*convert\_scale*= $\langle \text{x} \rangle$   $\langle \text{ESL} \rangle$

Options:

*activity* This determines which property (if any) to use as the activity property for the ligands.

Valid values: text strings

Default value:

*convert\_activity*

Set to true if the activity values should be converted from concentration to -log[concentration]

Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **false**

*convert\_scale*

A scale factor for conversion.

Valid values: reals

Default value: **1**

Operands:

$\langle \text{ESL} \rangle$

The entries to add as ligands.

## phaseaddligandsfromrun

Adds the given ligands to the table.

Syntax:

**phaseaddligandsfromrun** *runname*=⟨text⟩ ⟨ligand\_names⟩

Options:

*runname* This determines which run to use as the reference to copy the corresponding properties of the ligands.

Valid values: text strings

Default value:

Operands:

⟨ligand\_names⟩

The ligand names to be added separated by semi-colon.

## phasebuildqsar

Sets the QSAR options and launches a Build QSAR job.

Syntax:

**phasebuildqsar** *gridspacing*=⟨x⟩ *maxplsfactor*=⟨n⟩  
*modeltype*=atom\_based | phase\_based *tolerance\_a*=⟨x⟩  
*tolerance\_d*=⟨x⟩ *tolerance\_h*=⟨x⟩ *tolerance\_n*=⟨x⟩  
*tolerance\_p*=⟨x⟩ *tolerance\_r*=⟨x⟩ *tolerance\_x*=⟨x⟩  
*tolerance\_y*=⟨x⟩ *tolerance\_z*=⟨x⟩

Options:

*gridspacing*

The grid spacing to be for the QSAR model grid. The valid range is 0.5 to 2.0 angstrom.

Valid values: reals

Default value: 1

Minimum: 0.5

Maximum: 2.0

*maxplsfactor*

This option value range depends on the ligands in the training set. The valid range is 1 to N/5, where N is the number of ligands in the training set. If N is less than 5, the maximum number of factors is 1, but no QSAR model can be constructed.

Valid values: integers

Default value: **1**

Minimum: 1

*modeltype* The value of the model type in the Build QSAR Model step options dialog.

Valid values: atom\_based

phase\_based

Default value: **atom\_based**

*tolerance\_a*

The feature matching tolerance for the hydrogen bond acceptor feature. The valid range is 0.0 to 100.0

Valid values: reals

Default value: **1**

Minimum: 0.0

Maximum: 100.0

*tolerance\_d*

The feature matching tolerance for the hydrogen bond donor feature. The valid range is 0.0 to 100.0

Valid values: reals

Default value: **1**

Minimum: 0.0

Maximum: 100.0

*tolerance\_h*

The feature matching tolerance for the hydrophobic feature. The valid range is 0.0 to 100.0

Valid values: reals

Default value: **1.5**

Minimum: 0.0

Maximum: 100.0

*tolerance\_n*

The feature matching tolerance for the negative feature. The valid range is 0.0 to 100.0

Valid values: reals

Default value: **0.75**

Minimum: 0.0

Maximum: 100.0

*tolerance\_p*

The feature matching tolerance for the positive feature. The valid range is 0.0 to 100.0

Valid values: reals

Default value: **0.75**

Minimum: 0.0  
Maximum: 100.0

*tolerance\_r*

The feature matching tolerance for the aromatic ring feature.  
The valid range is 0.0 to 100.0

Valid values: reals  
Default value: **1.5**  
Minimum: 0.0  
Maximum: 100.0

*tolerance\_x*

The feature matching tolerance for the custom(X) feature. The  
valid range is 0.0 to 100.0

Valid values: reals  
Default value: **1**  
Minimum: 0.0  
Maximum: 100.0

*tolerance\_y*

The feature matching tolerance for the custom(Y) feature. The  
valid range is 0.0 to 100.0

Valid values: reals  
Default value: **1**  
Minimum: 0.0  
Maximum: 100.0

*tolerance\_z*

The feature matching tolerance for the custom(Z) feature. The  
valid range is 0.0 to 100.0

Valid values: reals  
Default value: **1**  
Minimum: 0.0  
Maximum: 100.0

## **phasechooseactiveset**

Sets the Active flag based on whether the activity value is larger than the  
cutoff value.

Syntax:

**phasechooseactiveset** < cutoff >

Operands:

< cutoff >

The cutoff value for deciding between active and inactive.

**phasescleanupstructures**

Launches a Cleanup Structures job for Phase.

Syntax:

**phasescleanupstructures**

**phaseclearcentroidatoms**

Removes all atoms from the centroid list for excluded volumes.

Syntax:

**phaseclearcentroidatoms**

**phaseconfgen**

Defines settings for Phase Generate Conformers job.

Syntax:

```
phaseconfgen field=mmffs | opls2005 incorporate=append |
    replace | ignore maxdist=<x> method=default | mixed
    numsteps=<n> postmaxiter=<n> postprocessing=yes | no
    premaxiter=<n> preprocessing=yes | no sampling=standard |
    rapid | complete | thorough solvation=gbsa |
    distance_dependent window=<x>
```

Options:

<i>field</i>	This determines which force field mmffs mmff opls2001 is used. Currently we always use mmffs, so it will have only one option value. Valid values: mmffs opls2005 Default value: <b>opls2005</b>
<i>incorporate</i>	This option controls the incorporation of the results (replace or append). Valid values: append replace ignore Default value: <b>replace</b>
<i>maxdist</i>	Maximum distance between atoms in equal structures. Valid values: reals Default value: <b>2</b> Minimum: 0.0
<i>method</i>	This determines whether MacroModel uses the ligand torsion search method (default) or the mixed MCMM/LMOD search method (mixed) to generate conformers. Valid values: default mixed Default value: <b>default</b>
<i>numsteps</i>	An option which sets the number of steps which will be performed during the conformational search. This also limits number of conformations generated. Valid values: integers Default value: <b>1000</b> Minimum: 0
<i>postmaxiter</i>	This option determines the maximum number of iterations for post-minimization of generated structures. Valid values: integers Default value: <b>50</b> Minimum: 0 Maximum: 9999999
<i>postprocessing</i>	Indicates whether or not to perform MacroModel postprocessing. Valid values: boolean (true false; yes no; y n; on off) Default value: <b>true</b>

*premaxiter*

This option determines the maximum number of iterations for pre-minimization of input structures.

Valid values: integers

Default value: **100**

Minimum: 0

Maximum: 9999999

*preprocessing*

Indicates whether or not to perform MacroModel preprocessing.

Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **true**

*sampling*

This determines whether rapid (standard) or thorough (complete) sampling will be used.

Valid values: standard

rapid

complete

thorough

Default value: **standard**

*solvation*

This determines whether GB/SA Water (gbsa) or Distance Dependent Dielectric (distance-dependent) solvation treatment is used.

Valid values: gbsa

distance\_dependent

Default value: **distance\_dependent**

*window*

The energy window ( in kcal/mol ) within which structures will be saved.

Valid values: reals

Default value: **10**

Minimum: 0.0

## **phasecreateexcludedvolume**

Creates an excluded volume from the current centroid atoms.

Syntax:

**phasecreateexcludedvolume**

**phasedbaddconfset**

Adds the selected confset to the Phase DB

Syntax:

**phasedbaddconfset**

**phasedbaddligands**

Adds ligands from the specified file to the database. The ligands will go through a cleanup process.

Syntax:

**phasedbaddligands** *alreadycleaned=yes | no ionization=retain | neutralize | generate maxisomers=<n> stereoisomers=retain | determine | all target\_ph=<x> <file name>*

Options:

*alreadycleaned*

Indicates whether or not the structures have already been cleaned.

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **false**

*ionization* Sets the handling of ionization.

Valid values: retain  
neutralize  
generate  
Default value: **retain**

*maxisomers*

The maximum number of stereoisomers to return for any ligand.

Valid values: integers  
Default value: **10**  
Minimum: 1

*stereoisomers*

Sets the handling of chiralities.

Valid values: retain  
determine  
all  
Default value: **all**

*target\_ph* The target pH for generating ionization states.

Valid values: reals  
Default value: **7**

Operands:

*<file name>*

The name of the file of structure to be added. These will go through the cleanup process with the options specified below. The ligands can be in Maestro or SD format.

## **phasedbaddligandsfromdb**

Adds ligands from the specified database

Syntax:

## **phasedbaddligandsfromdb**

## **phasedbconfigen**

Defines settings for PhaseDB Generate Conformers job.

Syntax:

```
phasedbconfig field=mmffs | opls2005 incorporate=append | replace | ignore maxdist=<x> method=default | mixed numsteps=<n> postmaxiter=<n> postprocessing=yes | no premaxiter=<n> preprocessing=yes | no sampling=standard | rapid | complete | thorough solvation=gbsa | distance-dependent window=<x>
```

Options:

<i>field</i>	This determines which force field mmffs mmff opls2001 is used. Currently we always use mmffs, so it will have only one option value.
Valid values:	mmffs opls2005
Default value:	<b>opls2005</b>
<i>incorporate</i>	This option controls the incorporation of the results (replace or append).
Valid values:	append replace ignore
Default value:	<b>replace</b>
<i>maxdist</i>	Maximum distance between atoms in equal structures.
Valid values:	reals
Default value:	<b>2</b>
Minimum:	0.0
<i>method</i>	This determines whether MacroModel uses the ligand torsion search method (default) or the mixed MCMM/LMOD search method (mixed) to generate conformers. Currently database creation always uses the default method, so it will have only one option value.
Valid values:	default mixed
Default value:	<b>default</b>
<i>numsteps</i>	An option which sets the number of steps which will be performed during the conformational search. This also limits number of conformations generated.
Valid values:	integers
Default value:	<b>100</b>
Minimum:	0
<i>postmaxiter</i>	This option determines the maximum number of iterations for post-minimization of generated structures.

Valid values: integers  
Default value: **50**  
Minimum: 0  
Maximum: 9999999

*postprocessing*

Indicates whether or not to perform MacroModel postprocessing.

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **false**

*premaxiter*

This option determines the maximum number of iterations for pre-minimization of input structures.

Valid values: integers  
Default value: **100**  
Minimum: 0  
Maximum: 9999999

*preprocessing*

Indicates whether or not to perform MacroModel preprocessing.

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **false**

*sampling*

This determines whether rapid (standard) or thorough (complete) sampling will be used.

Valid values: standard  
rapid  
complete  
thorough  
Default value: **standard**

*solvation*

This determines whether GB/SA Water (gbsa) or Distance Dependent Dielectric (distance\_dependent) solvation treatment is used.

Valid values: gbsa  
distance\_dependent  
Default value: **distance\_dependent**

*window*

The energy window ( in kcal/mol ) within which structures will be saved.

Valid values: reals  
Default value: **10**  
Minimum: 0.0

## **phasedbcreatesites**

Launches a Create Sites job for a Phase database.

Syntax:

**phasedbcreatesites** *withligands=all | withsites*

Options:

*withligands*

Controls which ligands have sites regenerated.

Valid values:      all  
                      withsites  
Default value:     **withsites**

## **phasedbcreatesubsetfromhitfile**

Create a new subset based on a Maestro structure file which is the result of a phase DB search.

Syntax:

**phasedbcreatesubsetfromhitfile** <subsetname> <hit file>

Operands:

<subsetname> <hit file>

The name of the new subset and a Maestro structure file which is the result of a Phase database search.

## **phasedbcreatesubsetfromselected**

Create a new subset from those ligands which are currently selected in the ligand table.

Syntax:

### **phasedbcreatesubsetfromselected** <subsetname>

Operands:

<subsetname>

The name of the new subset.

### **phasedbcreatesubsetfromtextfile**

Create a new subset based on a file containing ligand titles - one per line.

Syntax:

### **phasedbcreatesubsetfromtextfile** <subset name> <text file>

Operands:

<subset name> <text file>

The name of the subset and a file which contains a list of ligand names - one per line.

### **phasedbdeleteligands**

Launches a Delete Ligands job for selected ligands in a Phase DB.

Syntax:

### **phasedbdeleteligands** *whichstructures=selected | all*

Options:

*whichstructures*

Determines which structures this operation will be applied to.

Valid values: selected

all

Default value: **selected**

### **phasedbdeletesubset**

Delete the named subset from the DB

Syntax:

**phasedbdeletesubset** <subset name>

Operands:

<subset name>

The name of the subset to be deleted.

## **phasedbdisplayrange**

Control the range of ligands which are displayed in the table.

Syntax:

**phasedbdisplayrange** *end*=<n> *start*=<n>

Options:

*end* Sets the end point or the display range

Valid values: integers

Default value: **1000**

Minimum: 1

*start* Sets the starting point or the display range

Valid values: integers

Default value: 1

Minimum: 1

## **phasedbexportligands**

Export the ligands corresponding to selected ligands to the project table.

Syntax:

**phasedbexportligands** *confs*=firstconf | allconfs

Options:

*conf*s Controls what is exported - all conformations or just the first of each ligand.

Valid values:      firstconf  
                      allconfs  
Default value:     **firstconf**

## **phasedbfilter**

Replace currently filtered structures with ones which match the expression

Syntax:

**phasedbfilter** <filter\_expr>

Operands:

<filter\_expr>

The filter string which is to be applied to the ligand names to determine which ligands are to be displayed in the database table.

## **phasedbfilteradd**

Add to the currently filtered structures with ones which match the expression.

Syntax:

**phasedbfilteradd** <filter\_expr>

Operands:

<filter\_expr>

The filter string which is to be applied to the ligand names to determine which ligands are to be added to the database table.

## **phasedbgenerateconformers**

Launches a Generate Conformers job for a Phase database.

Syntax:

**phasedbgenerateconformers** *whichstructures=selected | all*

Options:

*whichstructures*

Determines which structures this operation will be applied to.

Valid values: selected

all

Default value: **selected**

## **phasedbnewfromfile**

Create a new Phase 3D database and open it in Maestro.

Syntax:

**phasedbnewfromfile** <db\_dir\_path> <name>

Operands:

<db\_dir\_path> <name>

The path (location) of the directory to be created as the new Phase 3D database directory and the name of the DB

## **phasedbopen**

Open an existing Phase 3D database into Maestro.

Syntax:

**phasedbopen** <db\_dir\_path> <db\_name>

Operands:

<db\_dir\_path> <db\_name>

The path (location) and name of the Phase 3D database directory to be opened.

## **phasedbremoveconformers**

Launches a Remove Conformers job for selected ligands in a Phase DB.

Syntax:

**phasedbremoveconformers** *whichstructures*=selected | all

Options:

*whichstructures*

Determines which structures this operation will be applied to.

Valid values: selected

all

Default value: **selected**

## **phasedbselectextendtablerow**

Extend the selection in the Phase Database table from the selected table row to joining up with an existing selection.

Syntax:

**phasedbselectextendtablerow** <row\_number>

Operands:

<row\_number>

The row number in the table from which the selection is to begin.

## **phasedbselectonlytablerow**

Select a row from the phase database table.

Syntax:

**phasedbselectonlytablerow** <row\_number>

Operands:

<row\_number>

The row number in the table which is to be selected.

### **phasedbselectsusubsetrow**

Select the row in the subset table.

Syntax:

**phasedbselectsusubsetrow <row>**

Operands:

<row>

The row number of the subset to be selected.

### **phasedbselecttablerow**

Selects a row from the phase database table.

Syntax:

**phasedbselecttablerow <row\_number>**

Operands:

<row\_number>

The row number in the table which is to be selected.

### **phasedbunselecttablerow**

Unselects a row from the Phase Database table.

Syntax:

**phasedbunselecttablerow <row\_number>**

Operands:

<row\_number>

The row number in the table which is to be unselected.

## **phasedeletecustomfeature**

Deletes the given custom feature.

Syntax:

**phasedeletecustomfeature** ⟨ code ⟩

Operands:

⟨ code ⟩

The code for the feature to delete.

## **phasedeleteexcludedvolumes**

Deletes the selected excluded volumes.

Syntax:

**phasedeleteexcludedvolumes**

## **phasedeletehypothesis**

Deletes the given hypothesis from the hypotheses table in the Score Hypotheses step.

Syntax:

**phasedeletehypothesis** ⟨ ID ⟩

Operands:

⟨ ID ⟩

The hypothesis to delete.

## **phasedeletehypothesisfrombuilder**

Deletes the given hypothesis from the hypotheses table in the Edit Hypothesis and Find Matches panels.

Syntax:

**phasedeletehypothesisfrombuilder** <row number>

Operands:

<row number>

The row number of the hypothesis to delete.

**phasedeleteselectedligands**

Deletes the selected ligands in the current Phase step.

Syntax:

**phasedeleteselectedligands**

**phasedisplayproperty**

Displays the given property from the table.

Syntax:

**phasedisplayproperty** <property name>

Operands:

<property name>

The property to display.

**phaseexcludetablerow**

Excludes the given row in the first table in the step from the Workspace.

Syntax:

**phaseexcludetablerow** <row>

Operands:

<row>

The row number to exclude in the Workspace.

**phaseexportalignmentsToFile**

Exports the selected alignments from the ligands table in the Score Hypotheses or Build QSAR steps.

Syntax:

**phaseexportalignmentsToFile** <file name>

Operands:

<file name>

The file name to export the selected alignments to.

**phaseexportconformersToFile**

Exports the selected conformers from the ligands table in the Prepare Ligands or Create Sites steps.

Syntax:

**phaseexportconformersToFile** <file name>

Operands:

<file name>

The file name to export the selected conformers to.

**phaseexportfeature**

Exports the features to the given file name.

Syntax:

**phaseexportfeature** <file name>

Operands:

<file name>

A path to a write the features file to.

**phaseexporthypothesis**

Exports the selected hypothesis from the hypotheses table in the Score Hypotheses step.

Syntax:

**phaseexporthypothesis** <file name>

Operands:

<file name>

The file name to export the selected hypothesis to.

**phaseexporthypothesisfrombuilder**

Exports the selected hypothesis from the hypotheses table in the Edit Hypotheses panel.

Syntax:

**phaseexporthypothesisfrombuilder** <file name>

Operands:

<file name>

The file name to export the selected hypothesis to.

## phaseexportselectedalignments

Exports the selected alignments from the current Phase step to the project table.

Syntax:

### phaseexportselectedalignments

## phaseexportselectedligands

Exports the selected ligands from the current Phase step to the project table.

Syntax:

### phaseexportselectedligands

## phasefindmatches

Launches a Find Matches job.

Syntax:

```
phasefindmatches align_cutoff=<x> align_weight=<x>
apply_excluded_volumes=yes | no conformers=existing |
generate database=<text> distance_tolerance=<x>
extfile=<text> hits_molecule=<n> hits_total=<n>
jobinput=extfile | selectedentries | 3ddatabase matches=new |
existing minsites=<n> subset=<text> useqsar=yes | no
vector_weight=<x> volume_weight=<x> <job_id> | <job_name>
| <job_name> <project_name>
```

Options:

### *align\_cutoff*

The weight for the alignment score.

Valid values:    reals

Default value:    **1.2**

Minimum:        0.0001

*align\_weight*

The weight for the alignment score

Valid values: reals

Default value: **1**

Minimum: 0.0

*apply\_excluded\_volumes*

Indicates whether or not to use the excluded volumes information.

Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **true**

*conformers*

Whether to use existing conformers or generate them during the search

Valid values: existing

generate

Default value: **generate**

*database*

The source database file for Phase Find Matches.

Valid values: text strings

Default value:

*distance\_tolerance*

Valid values: reals

Default value: **2**

*extfile*

The source structure .sd or .mae file for Phase Find Matches.

Valid values: text strings

Default value:

*hits\_molecule*

The maximum number of hits per molecule to return.

Valid values: integers

Default value: **1**

Minimum: 1

*hits\_total*

The maximum number of hits to return.

Valid values: integers

Default value: **1000**

Minimum: 1

*jobinput*

The source of job input of Phase Find Matches.

Valid values: extfile

selectedentries

3ddatabase

Default value: **3ddatabase**

<i>matches</i>	Whether to search for new or existing matches.
Valid values:	new existing
Default value:	<b>new</b>
<i>minsites</i>	The minimum number of sites to match.
Valid values:	integers
Default value:	<b>5</b>
Minimum:	1
<i>subset</i>	The name of the subset for the Phase Find Matches.
Valid values:	text strings
Default value:	
<i>useqsar</i>	The flag of using QSAR model or not.
Valid values:	boolean (true false; yes no; y n; on off)
Default value:	<b>true</b>
<i>vector_weight</i>	The weight for the vector score.
Valid values:	reals
Default value:	<b>1</b>
Minimum:	0.0
<i>volume_weight</i>	The weight for the volume score.
Valid values:	reals
Default value:	<b>1</b>
Minimum:	0.0

Operands:

$\langle \text{job\_id} \rangle \mid \langle \text{job\_name} \rangle \mid \langle \text{job\_name} \rangle \langle \text{project\_name} \rangle$

The ID or name of the Find Matches job which is to be run.

## **phaselinefindpharmacophores**

Sets the Find Common Pharmacophores options and launches the job.

Syntax:

**phasefindpharmacophores** *finalboxsize=⟨x⟩ maxtreedepth=⟨n⟩ minintersitedistance=⟨x⟩*

Options:

*finalboxsize*

The final box size in angstrong.

Valid values:   reals

Default value:   **1**

*maxtreedepth*

The maximum depth of the tree.

Valid values:   integers

Default value:   **4**

*minintersitedistance*

The minimum distance between the sites in angstrong.

Valid values:   reals

Default value:   **2**

## **phasegenerateconformers**

Launches a Generate Conformers job for Phase.

Syntax:

## **phasegenerateconformers**

## **phasehookimport**

Sets up the Import panel and dialog for Phase to import ligands and conformers.

Syntax:

**phasehookimport**

**phasehypothesiscreateexcludedvolume**

Creates an excluded volume from the current centroid atoms for the currently selected hypothesis.

Syntax:

**phasehypothesiscreateexcludedvolume**

**phasehypothesisdeleteexcludedvolumes**

Deletes the selected excluded volumes for the currently selected hypothesis.

Syntax:

**phasehypothesisdeleteexcludedvolumes**

**phasehypothesiselectevrow**

Selects the given row in the excluded volumes table for the currently selected hypothesis.

Syntax:

**phasehypothesiselectevrow** <row>

Operands:

<row>

The row number to select in the table.

**phasehypothesiselectextendevrow**

Extends the selection to this row in the excluded volumes table for the currently selected hypothesis.

Syntax:

**phasehypothesisselectextendrow** <row>

Operands:

<row>

The row number to extend the select to.

### **phasehypothesisselectonlyevrow**

Selects only this row in the excluded volumes table for the currently selected hypothesis.

Syntax:

**phasehypothesisselectonlyevrow** <row>

Operands:

<row>

The row number to select only in the table row.

### **phasehypothesisselectrow**

Toggles the given hypothesis into or out of the Workspace based on the currently selected hypothesis.

Syntax:

**phasehypothesisselectrow** <row>

Operands:

<row>

The row number to toggle inclusion state for.

### **phasehypothesissetexcludedvolumes**

Sets the value for the given cell for the currently selected hypothesis.

Syntax:

**phasehypothesissetexcludedvolumes** *column=* $\langle n \rangle$  *row=* $\langle n \rangle$   
 $\langle$  *value*  $\rangle$

Options:

*column* The column number of the cell to change.

Valid values: integers

Default value: 1

*row* The row number of the cell to change.

Valid values: integers

Default value: 1

Operands:

$\langle$  *value*  $\rangle$

The value for the given cell.

## **phasehypothesissetsitemask**

Sets the sitemask for the currently selected hypothesis

Syntax:

**phasehypothesissetsitemask**  $\langle$  *value*  $\rangle$

Operands:

$\langle$  *value*  $\rangle$

The value for the given cell.

## **phasehypothesissettolerances**

Sets the feature matching tolerances for the currently selected hypothesis

Syntax:

**phasehypthesissettolerances** *tolerance\_a=⟨x⟩*  
*tolerance\_d=⟨x⟩ tolerance\_h=⟨x⟩ tolerance\_n=⟨x⟩*  
*tolerance\_p=⟨x⟩ tolerance\_r=⟨x⟩ tolerance\_x=⟨x⟩*  
*tolerance\_y=⟨x⟩ tolerance\_z=⟨x⟩ usetolerances=yes | no*

Options:

*tolerance\_a*

The feature matching tolerance for the hydrogen bond acceptor feature. The valid range is 0.0 to 100.0

Valid values: reals

Default value: **1**

Minimum: 0.0

Maximum: 100.0

*tolerance\_d*

The feature matching tolerance for the hydrogen bond donor feature. The valid range is 0.0 to 100.0

Valid values: reals

Default value: **1**

Minimum: 0.0

Maximum: 100.0

*tolerance\_h*

The feature matching tolerance for the hydrophobic feature. The valid range is 0.0 to 100.0

Valid values: reals

Default value: **1.5**

Minimum: 0.0

Maximum: 100.0

*tolerance\_n*

The feature matching tolerance for the negative feature. The valid range is 0.0 to 100.0

Valid values: reals

Default value: **0.75**

Minimum: 0.0

Maximum: 100.0

*tolerance\_p*

The feature matching tolerance for the positive feature. The valid range is 0.0 to 100.0

Valid values: reals

Default value: **0.75**

Minimum: 0.0

Maximum: 100.0

*tolerance\_r*

The feature matching tolerance for the aromatic ring feature.  
The valid range is 0.0 to 100.0

Valid values:    reals  
Default value: **1.5**  
Minimum:       0.0  
Maximum:      100.0

*tolerance\_x*

The feature matching tolerance for the custom(X) feature. The  
valid range is 0.0 to 100.0

Valid values:    reals  
Default value: **1**  
Minimum:       0.0  
Maximum:      100.0

*tolerance\_y*

The feature matching tolerance for the custom(Y) feature. The  
valid range is 0.0 to 100.0

Valid values:    reals  
Default value: **1**  
Minimum:       0.0  
Maximum:      100.0

*tolerance\_z*

The feature matching tolerance for the custom(Z) feature. The  
valid range is 0.0 to 100.0

Valid values:    reals  
Default value: **1**  
Minimum:       0.0  
Maximum:      100.0

*usetolerances*

When using feature matching tolerances, this option is true otherwise false.

Valid values:    boolean (true|false; yes|no; y|n; on|off)  
Default value: **false**

## **phasehypothesistoggleinclude**

Toggles the given hypothesis into or out of the Workspace.

Syntax:

**phasehypothesistoggleinclude <row>**

Operands:

<row>

The row number to toggle inclusion state for.

**phasehypothesisunselectevrow**

Unselects the given row in the excluded volumes table for the currently selected hypothesis.

Syntax:

**phasehypothesisunselectevrow <row>**

Operands:

<row>

The row number to unselect in the table.

**phaseimportfeature**

Imports the features from the given file name. This replaces the existing features.

Syntax:

**phaseimportfeature <file name>**

Operands:

<file name>

A path to a valid features file.

## **phaseimporthypothesis**

Imports a hypothesis from the selected file into the Find Matches and Edit Hypotheses panels.

Syntax:

**phaseimporthypothesis** *isphase=yes | no* <file name>

Options:

*isphase* Indicates whether or not the hypothesis is a Phase hypothesis or an external hypothesis.

Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: false

Operands:

<file name>

The file name to import a hypothesis from.

## **phaseincludeextendtablerow**

Extends the rows included in the workspace to include this one.

Syntax:

**phaseincludeextendtablerow** <row>

Operands:

<row>

The row number to include in the Workspace.

## **phaseincludeonlytablerow**

Includes only the given row in the first table in the step into the Workspace.

Syntax:

**phaseincludeonlytablerow** <row>

Operands:

<row>

The row number to include in the Workspace.

**phaseincludetablerow**

Includes the given row in the first table in the step into the Workspace.

Syntax:

**phaseincludetablerow** <row>

Operands:

<row>

The row number to include in the Workspace.

**phaseinverttableselection**

Inverts the row selection in the first table in the step.

Syntax:

**phaseinverttableselection**

**phaseligprep**

Defines settings for Phase LigPrep job.

Syntax:

**phaseligprep** *gen\_stereo*=<n> *ionization*=generate | neutralize |  
retain *ph*=<x> *stereoisomers*=retain | determine | generate

Options:

*gen\_stereo* Generate stereoisomers (maximum)

Valid values: integers  
Default value: **32**  
Minimum: 1

*ionization* How to generate the ionization states

Valid values: generate  
neutralize  
retain  
Default value: **retain**

*ph* The ionization pH

Valid values: reals  
Default value: **7**  
Minimum: 0.0

*stereoisomers*

How to generate the stereoisomers

Valid values: retain  
determine  
generate  
Default value: **retain**

## **phaselocatesites**

Launches a Locate Sites job.

Syntax:

## **phaselocatesites**

## **phasemarkerdump**

Print out the current option values of the phase marker command.

Syntax:

**phasemarkerdump****phasemarkersettings**

Set graphical data of feature markers.

Syntax:

```
phasemarkersettings acceptorarrowblue=<x>
acceptorarrowgreen=<x> acceptorarrowred=<x>
acceptorcornradius=<x> acceptorcylinderheight=<x>
acceptorcylinderradius=<x> acceptorsiteblue=<x>
acceptorsitegreen=<x> acceptorsiteradius=<x>
acceptorsitered=<x> ambient=<x> aromaticringradius=<x>
aromaticringradius5=<x> aromaticsiteblue=<x>
aromaticsitegreen=<x> aromaticsitered=<x>
aromatictuberadius=<x> aromaticustep=<n>
aromaticvstep=<n> diffuse=<x> donorarrowblue=<x>
donorarrowgreen=<x> donorarrowred=<x>
donorcornradius=<x> donorcylinderheight=<x>
donorcylinderradius=<x> donorsiteblue=<x>
donorsitegreen=<x> donorsiteradius=<x> donorsitered=<x>
drawstyle=solid | line emission=<x> featurexsiteblue=<x>
featurexsitegreen=<x> featurexsiteradius=<x>
featurexsitered=<x> featureysiteblue=<x>
featureysitegreen=<x> featureysiteradius=<x>
featureysitered=<x> featurezsiteblue=<x>
featurezsitegreen=<x> featurezsiteradius=<x>
featurezsitered=<x> hydrophobicssiteblue=<x>
hydrophobicssitegreen=<x> hydrophobicsiteradius=<x>
hydrophobicsitered=<x> linewidth=<n> negativesiteblue=<x>
negativesitegreen=<x> negativesiteradius=<x>
negativesitered=<x> positivesiteblue=<x> positivesitegreen=<x>
positivesiteradius=<x> positivesitered=<x> shininess=<x>
sliceline=<n> slicesolid=<n> specular=<x> stackline=<n>
stacksolid=<n> transparency=<x>
```

Options:

*acceptorarrowblue*

The arrow color blue component of acceptor markers.

Valid values:    reals

Default value:    **0.5**

Minimum: 0.0  
Maximum: 1.0

*acceptorarrowgreen*

The arrow color green component of acceptor markers.

Valid values: reals  
Default value: **0.5**  
Minimum: 0.0  
Maximum: 1.0

*acceptorarrowred*

The arrow color red component of acceptor markers.

Valid values: reals  
Default value: **1**  
Minimum: 0.0  
Maximum: 1.0

*acceptorcornradius*

The radius of arrow corn of acceptor markers.

Valid values: reals  
Default value: **0.35**  
Minimum: 0.0

*acceptorcylinderheight*

The height of arrow cylinder of acceptor markers.

Valid values: reals  
Default value: **0.8**  
Minimum: 0.0  
Maximum: 1.0

*acceptorcylinderradius*

The radius of arrow cylinder of acceptor markers.

Valid values: reals  
Default value: **0.15**  
Minimum: 0.0

*acceptorsiteblue*

The site color blue component of acceptor markers.

Valid values: reals  
Default value: **0.5**  
Minimum: 0.0  
Maximum: 1.0

*acceptorsitegreen*

The site color green component of acceptor markers.

Valid values: reals  
Default value: **0.5**

Minimum: 0.0  
Maximum: 1.0

*acceptorsiteradius*

The radius of acceptor site markers.

Valid values: reals  
Default value: **0.8**  
Minimum: 0.0

*acceptorsitered*

The site color red component of acceptor markers.

Valid values: reals  
Default value: **1**  
Minimum: 0.0  
Maximum: 1.0

*ambient*

Set material property - ambient, to its red, green, and blue components, for front face.

Valid values: reals  
Default value: **0.5**  
Minimum: 0.0  
Maximum: 1.0

*aromaticringradius*

The radius of the ring of the ring torus, for the ring constructed by 6 or more atoms.

Valid values: reals  
Default value: **0.5**  
Minimum: 0.0

*aromaticringradius5*

The radius of the ring of the ring torus, for the ring constructed by 5 or less atoms.

Valid values: reals  
Default value: **0.4**  
Minimum: 0.0

*aromaticsiteblue*

The site color blue component of aromatic markers.

Valid values: reals  
Default value: **0.2**  
Minimum: 0.0  
Maximum: 1.0

*aromaticsitegreen*

The site color green component of aromatic markers.

Valid values: reals

Default value: **0.51**  
Minimum: 0.0  
Maximum: 1.0

*aromaticsitered*

The site color red component of aromatic markers.

Valid values: reals  
Default value: **0.92**  
Minimum: 0.0  
Maximum: 1.0

*aromatictuberadius*

The radius of the tube of the ring torus.

Valid values: reals  
Default value: **0.15**  
Minimum: 0.0

*aromaticustep*

The U step domain tolerance.

Valid values: integers  
Default value: **5**  
Minimum: 2

*aromaticvstep*

The V step domain tolerance.

Valid values: integers  
Default value: **3**  
Minimum: 2

*diffuse*

Set material property - diffuse, to its red, green, and blue components, for front face.

Valid values: reals  
Default value: **0.4**  
Minimum: 0.0  
Maximum: 1.0

*donorarrowblue*

The arrow color blue component of donor markers.

Valid values: reals  
Default value: **0.9**  
Minimum: 0.0  
Maximum: 1.0

*donorarrowgreen*

The arrow color green component of donor markers.

Valid values: reals  
Default value: **0.8**

Minimum: 0.0  
Maximum: 1.0

*donorarrowred*

The arrow color red component of donor markers.

Valid values: reals  
Default value: **0.5**  
Minimum: 0.0  
Maximum: 1.0

*donorcornradius*

The radius of arrow corn of donor markers.

Valid values: reals  
Default value: **0.35**  
Minimum: 0.0

*donorcylinderheight*

The height of arrow cylinder of donor markers.

Valid values: reals  
Default value: **0.8**  
Minimum: 0.0  
Maximum: 1.0

*donorcylinderradius*

The radius of arrow cylinder of donor markers.

Valid values: reals  
Default value: **0.15**  
Minimum: 0.0

*donorsiteblue*

The site color blue component of donor markers.

Valid values: reals  
Default value: **0.9**  
Minimum: 0.0  
Maximum: 1.0

*donorsitegreen*

The site color green component of donor markers.

Valid values: reals  
Default value: **0.8**  
Minimum: 0.0  
Maximum: 1.0

*donorsiteradius*

The radius of donor site markers.

Valid values: reals  
Default value: **0.8**  
Minimum: 0.0

*donorsitered*

The site color red component of donor markers.

Valid values:   reals  
Default value: **0.5**  
Minimum:       0.0  
Maximum:      1.0

*drawstyle*   The styles of rendering feature markers, they are: 1 - solid, and 2 - lines. Default is solid.

Valid values:   solid  
                 line  
Default value: **solid**

*emission*   Set material property - emission, to its red, green, and blue components, for front face.

Valid values:   reals  
Default value: **0.05**  
Minimum:       0.0  
Maximum:      1.0

*featurexsiteblue*

The site color blue component of featurex markers.

Valid values:   reals  
Default value: **1**  
Minimum:       0.0  
Maximum:      1.0

*featurexsitegreen*

The site color green component of featurex markers.

Valid values:   reals  
Default value: **1**  
Minimum:       0.0  
Maximum:      1.0

*featurexsiteradius*

The radius of featurex site markers.

Valid values:   reals  
Default value: **0.8**  
Minimum:       0.0

*featurexsitered*

The site color red component of featurex markers.

Valid values:   reals  
Default value: **0**  
Minimum:       0.0  
Maximum:      1.0

*featureysiteblue*

The site color blue component of featurey markers.

Valid values:   reals

Default value:   **0**

Minimum:       0.0

Maximum:       1.0

*featureysitegreen*

The site color green component of featurey markers.

Valid values:   reals

Default value:   **1**

Minimum:       0.0

Maximum:       1.0

*featureysiteradius*

The radius of featurey site markers.

Valid values:   reals

Default value:   **0.8**

Minimum:       0.0

*featureysitered*

The site color red component of featurey markers.

Valid values:   reals

Default value:   **1**

Minimum:       0.0

Maximum:       1.0

*featurezsiteblue*

The site color blue component of featurez markers.

Valid values:   reals

Default value:   **1**

Minimum:       0.0

Maximum:       1.0

*featurezsitegreen*

The site color green component of featurez markers.

Valid values:   reals

Default value:   **0**

Minimum:       0.0

Maximum:       1.0

*featurezsiteradius*

The radius of featurez site markers.

Valid values:   reals

Default value:   **0.8**

Minimum:       0.0

*featurezsitered*

The site color red component of featurez markers.

Valid values:   reals  
Default value: **1**  
Minimum:       0.0  
Maximum:      1.0

*hydrophobicssiteblue*

The site color blue component of hydrophobic markers.

Valid values:   reals  
Default value: **0.3**  
Minimum:       0.0  
Maximum:      1.0

*hydrophobicssitegreen*

The site color green component of hydrophobic markers.

Valid values:   reals  
Default value: **1**  
Minimum:       0.0  
Maximum:      1.0

*hydrophobicsiteradius*

The radius of hydrophobic site markers.

Valid values:   reals  
Default value: **0.8**  
Minimum:       0.0

*hydrophobicssitered*

The site color red component of hydrophobic markers.

Valid values:   reals  
Default value: **0.3**  
Minimum:       0.0  
Maximum:      1.0

*linewidth*   Set the width of lines in drawing sphere.

Valid values:   integers  
Default value: **2**  
Minimum:       1

*negativesiteblue*

The site color blue component of negative markers.

Valid values:   reals  
Default value: **0.3**  
Minimum:       0.0  
Maximum:      1.0

*negativesitegreen*

The site color green component of negative markers.

Valid values:   reals  
Default value: **0.3**  
Minimum:       0.0  
Maximum:      1.0

*negativesiteradius*

The radius of negative site markers.

Valid values:   reals  
Default value: **0.8**  
Minimum:       0.0

*negativesitered*

The site color red component of negative markers.

Valid values:   reals  
Default value: **1**  
Minimum:       0.0  
Maximum:      1.0

*positivesiteblue*

The site color blue component of positive markers.

Valid values:   reals  
Default value: **1**  
Minimum:       0.0  
Maximum:      1.0

*positivesitegreen*

The site color green component of positive markers.

Valid values:   reals  
Default value: **0.6**  
Minimum:       0.0  
Maximum:      1.0

*positivesiteradius*

The radius of positive site markers.

Valid values:   reals  
Default value: **0.8**  
Minimum:       0.0

*positivesitered*

The site color red component of positive markers.

Valid values:   reals  
Default value: **0.5**  
Minimum:       0.0  
Maximum:      1.0

<i>shininess</i>	Set material property - shininess, for front face. Valid values:   reals Default value: <b>80</b> Minimum:       0.0 Maximum:      128.0
<i>sliceline</i>	Set the slices of drawing line sphere. Valid values:   integers Default value: <b>10</b> Minimum:       2
<i>slicesolid</i>	Set the slices of drawing solid sphere. Valid values:   integers Default value: <b>36</b> Minimum:       2
<i>specular</i>	Set material property - specular, to its red, green, and blue components, for front face. Valid values:   reals Default value: <b>0</b> Minimum:       0.0 Maximum:      1.0
<i>stackline</i>	Set the stacks of drawing line sphere. Valid values:   integers Default value: <b>8</b> Minimum:       2
<i>stacksolid</i>	Set the stacks of drawing solid sphere. Valid values:   integers Default value: <b>18</b> Minimum:       2
<i>transparency</i>	The transparency of rendering feature markers. Valid values:   reals Default value: <b>20</b> Minimum:       0.0 Maximum:      100.0

## phasemarkfeature

Marks the given feature in the Workspace.

Syntax:

**phasemarkfeature** <feature>

Operands:

<feature>

A single letter (A-Z) indicating the feature to mark.

## **phaseoptions**

Sets the options for Phase.

Syntax:

**phaseoptions** *random\_seed*=<n> *separate\_stereoisomers*=yes | no

Options:

*random\_seed*

The seed used to compute the random training / test set in the Build QSAR step. Zero means to use a completely random seed—any other value is used explicitly.

Valid values: integers

Default value: **0**

Minimum: 0

*separate\_stereoisomers*

Whether to separate stereoisomers or combine them into a single confset.

Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **true**

## **phasepatterndelete**

Deletes the given pattern.

Syntax:

## **phaseratterndelete** *pattern=⟨text⟩ ⟨feature⟩*

Options:

*pattern* The SMARTS pattern to remove.

Valid values: text strings

Default value:

Operands:

⟨feature⟩

The feature to remove the pattern from.

## **phaseratternedit**

Updates the given pattern with the new data.

Syntax:

```
phaseratternedit geometry=group | point | vector group=⟨text⟩  
          group_all=yes | no pattern=⟨text⟩ point=⟨n⟩ vector=⟨n⟩  
          ⟨feature⟩
```

Options:

*geometry* The type of geometry for this pattern.

Valid values: group  
point  
vector

Default value: **group**

*group* The atom numbers for the pattern if it is group geometry.

Valid values: text strings

Default value:

*group\_all* Set to true if the group should include all atom numbers.

Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **true**

*pattern* The SMARTS pattern to update.

Valid values: text strings

Default value:

*point* The atom number for the pattern if it is point geometry.

Valid values: integers

Default value: **1**  
Minimum: 1

*vector* The atom number for the pattern if it is vector geometry.  
Valid values: integers  
Default value: **1**  
Minimum: 1

Operands:  
 $\langle \text{feature} \rangle$   
The feature to which the pattern is associated with.

## **phaselinepatternmovedown**

Moves the given pattern down lower in the list for the feature.

Syntax:

**phaselinepatternmovedown** *pattern*= $\langle \text{text} \rangle$   $\langle \text{feature} \rangle$

Options:

*pattern* The SMARTS pattern to move.  
Valid values: text strings  
Default value:

Operands:  
 $\langle \text{feature} \rangle$   
The feature which contains the pattern.

## **phaselinepatternmoveup**

Moves the given pattern up higher in the list for the feature.

Syntax:

### **phaseratternmoveup** *pattern=⟨text⟩ ⟨feature⟩*

Options:

*pattern*      The SMARTS pattern to move.  
Valid values:    text strings  
Default value:

Operands:

⟨feature⟩  
The feature which contains the pattern.

### **phaseratternnew**

Adds the given pattern.

Syntax:

```
phaseratternnew geometry=group | point | vector group=⟨text⟩  
                  group_all=yes | no pattern=⟨text⟩ point=⟨n⟩ vector=⟨n⟩  
                  ⟨feature⟩
```

Options:

*geometry*      The type of geometry for this pattern.  
Valid values:    group  
                  point  
                  vector  
Default value:    **group**  
*group*          The atom numbers for the pattern if it is group geometry.  
Valid values:    text strings  
Default value:  
*group\_all*     Set to true if the group should include all atom numbers.  
Valid values:    boolean (true|false; yes|no; y|n; on|off)  
Default value:    **true**  
*pattern*       The SMARTS pattern to add.  
Valid values:    text strings  
Default value:  
*point*          The atom number for the pattern if it is point geometry.  
Valid values:    integers

Default value: **1**  
 Minimum: 1

*vector* The atom number for the pattern if it is vector geometry.  
 Valid values: integers  
 Default value: **1**  
 Minimum: 1

Operands:  
 ⟨feature⟩  
 The feature to add the pattern to.

## phaselineoptions

Sets the options for the given pattern.

Syntax:

**phaselineoptions** *exclude=yes | no ignore=yes | no mark=yes | no pattern=⟨text⟩ ⟨feature⟩*

Options:

*exclude* Indicates whether or not to exclude the pattern from the feature.  
 Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **true**

*ignore* Indicates whether or not to ignore the pattern in the feature.  
 Valid values: boolean (true|false; yes|no; y|n; on|off)  
 Default value: **true**

*mark* Indicates whether or not to mark the pattern in the Workspace.  
 Valid values: boolean (true|false; yes|no; y|n; on|off)  
 Default value: **true**

*pattern* The SMARTS pattern to update.  
 Valid values: text strings  
 Default value:

Operands:  
 ⟨feature⟩  
 The feature which contains the pattern.

## **phasepreview**

Launches a Preview job.

Syntax:

### **phasepreview**

## **phaseqsarsearch**

Transfers the current hypothesis from the Build QSAR step to the Find Matches panel. Also opens the Find Matches panel. Deprecated - use phaseresearchformatches instead.

Syntax:

### **phaseqsarsearch**

## **phaseqsarselecthypothesis**

Selects the given hypothesis.

Syntax:

### **phaseqsarselecthypothesis <ID>**

Operands:

<ID>

The ID of the hypothesis to select in the Build QSAR step. This will populate the Alignment table. This function is single select. If the operand is blank, the alignments table will be filled in for the currently selected hypothesis, if any.

## **phaserandomtraining**

Sets the random training set.

Syntax:

### **phaserandomtraining**

### **phaserandomtrainingoptions**

Holds the options for the random training set.

Syntax:

#### **phaserandomtrainingoptions** *percentage=⟨n⟩*

Options:

*percentage* The percentage of ligands to use as a training set.

Valid values: integers

Default value: **50**

Minimum: 1

Maximum: 100

### **phaserefreshfrequencytable**

Restores the default set of min/max values to the feature frequencies table in the Find Common Pharmacophores step.

Syntax:

### **phaserefreshfrequencytable**

### **phaseremoveligands**

Removes the given ligands from the table.

Syntax:

### **phaseremoveligands** <ESL>

Operands:

<ESL>

The entries to remove as ligands.

### **phaserenameligand**

Changes the name for the given ligand.

Syntax:

**phaserenameligand** *row*=<n> <new name>

Options:

*row* The row number of the ligand to rename.

Valid values: integers

Default value: 1

Minimum: 1

Operands:

<new name>

The new name for the given ligand.

### **phaserescorehypotheses**

Rescores all the hypotheses in the hypotheses table in the Score Hypotheses step.

Syntax:

**phaserescorehypotheses**

### **phaserescoreweighting**

Sets the weighting factors for rescored hypotheses in the Score Hypotheses step.

Syntax:

```
phaserescoreweighting activity=⟨x⟩ energy=⟨x⟩ inactive=⟨x⟩
match=⟨x⟩ selectivity=⟨x⟩ site=⟨x⟩ vector=⟨x⟩
volume=⟨x⟩
```

Options:

<i>activity</i>	The reference ligand activity scoring factor. The valid range is 0.0 to 100.0
	Valid values:   reals
	Default value: <b>0</b>
	Minimum:       0.0
	Maximum:      100.0
<i>energy</i>	The reference ligand relative conformational energy scoring factor. The valid range is 0.0 to 100.0
	Valid values:   reals
	Default value: <b>0</b>
	Minimum:       0.0
	Maximum:      100.0
<i>inactive</i>	The inactive match scoring factor. The valid range is 0.0 to 100.0
	Valid values:   reals
	Default value: <b>1</b>
	Minimum:       0.0
	Maximum:      100.0
<i>match</i>	The number of matches scoring factor. The valid range is 1.0 to infinity, but values only slightly above 1.0 might generate huge values for a large number of matched ligands.
	Valid values:   reals
	Default value: <b>1</b>
	Minimum:       1.0
<i>selectivity</i>	The selectivity scoring factor. The valid range is 0.0 to 100.0
	Valid values:   reals
	Default value: <b>0</b>
	Minimum:       0.0
	Maximum:      100.0
<i>site</i>	The aligned sites scoring factor. The valid range is 0.0 to 100.0
	Valid values:   reals
	Default value: <b>1</b>
	Minimum:       0.0
	Maximum:      100.0

*vector*      The vector scoring factor. The valid range is 0.0 to 100.0

Valid values:    reals

Default value:    1

Minimum:        0.0

Maximum:        100.0

*volume*      The volume scoring factor. The valid range is 0.0 to 100.0

Valid values:    reals

Default value:    1

Minimum:        0.0

Maximum:        100.0

## **phaseresetfeatures**

Resets the features to the installation defaults.

Syntax:

### **phaseresetfeatures**

## **phaseruncreate**

Creates the run with the given name.

Syntax:

### **phaseruncreate** <run name>

Operands:

<run name>

The name of the new run to create.

## **phaserundelete**

Deletes the current run from the project.

Syntax:

**phaserundele**t

**phaserunopen**

Opens the run with the given name.

Syntax:

**phaserunopen** <run name>

Operands:

<run name>

The name of the run to open.

**phaserunrename**

Changes the current run's name to the given name.

Syntax:

**phaserunrename** <run name>

Operands:

<run name>

The name to change the current run's name to.

**phaserunsaveas**

Saves a copy of the current run under the given name.

Syntax:

**phaserunsaveas** <run name>

Operands:

<run name>

The name of the run to save as.

**phaserunsetseed**

Sets the random seed of the current run based on the current Phase options.

Syntax:

**phaserunsetseed**

**phaserunsetstereo**

Sets the stereoisomer behavior of the current run based on the current Phase options.

Syntax:

**phaserunsetstereo**

**phasescorehypotheses**

Launches a Score Hypotheses job.

Syntax:

**phasescorehypotheses** *activityweight*=⟨x⟩ *aligncutoff*=⟨x⟩  
*energyweight*=⟨x⟩ *matchreward*=⟨x⟩ *maxboxes*=⟨n⟩  
*minboxes*=⟨n⟩ *selectivitybasedalignment*=yes | no  
*selectivityweight*=⟨x⟩ *sitetoppercentage*=⟨n⟩ *siteweight*=⟨x⟩  
*tolerance\_a*=⟨x⟩ *tolerance\_d*=⟨x⟩ *tolerance\_h*=⟨x⟩  
*tolerance\_n*=⟨x⟩ *tolerance\_p*=⟨x⟩ *tolerance\_r*=⟨x⟩  
*tolerance\_x*=⟨x⟩ *tolerance\_y*=⟨x⟩ *tolerance\_z*=⟨x⟩  
*usetolerances*=yes | no *vectorlowercutoff*=⟨x⟩  
*vectorweight*=⟨x⟩ *volumebasedalignment*=yes | no  
*volumeweight*=⟨x⟩

Options:

*activityweight*

For calculating the Survival score, the weighting factor of the reference ligand activity. The valid range is 0.0 to 100.0

Valid values:   reals

Default value:   **0**

Minimum:       0.0

Maximum:      100.0

*aligncutoff*

In case of score by site-based alignment, keep the hypotheses with RMSD values below the value specified by this option. The valid range is 0.0001 to infinity

Valid values:   reals

Default value:   **1.2**

Minimum:       0.0001

*energyweight*

For calculating the Survival score, the weighting factor of the reference ligand relative conformational energy. The valid range is 0.0 to 100.0

Valid values:   reals

Default value:   **0**

Minimum:       0.0

Maximum:      100.0

*matchreward*

For calculating the Survival score, increase the score by this value, raised to the power of the number of matched ligands. The valid range is 1.0 to infinity, but values only slightly above 1.0 might generate huge values for a large number of matched ligands.

Valid values:   reals

Default value:   **1**

	Minimum:	1.0
<i>maxboxes</i>	In case of score by site-based alignment, keep at most the number specified by this option. The valid range is 1 to infinity	
	Valid values:	integers
	Default value:	<b>50</b>
	Minimum:	1
<i>minboxes</i>	In case of score by site-based alignment, keep at least the number specified by this option. The valid range is 1 to infinity	
	Valid values:	integers
	Default value:	<b>10</b>
	Minimum:	1
<i>selectivitybasedalignment</i>		
	For score by selectivity-based alignment, this option is true otherwise false.	
	Valid values:	boolean (true false; yes no; y n; on off)
	Default value:	<b>true</b>
<i>selectivityweight</i>		
	For calculating the Survival score, the weighting factor of the selectivity score. The valid range is 0.0 to 100.0	
	Valid values:	reals
	Default value:	<b>0</b>
	Minimum:	0.0
	Maximum:	100.0
<i>sitetoppercentage</i>		
	In case of score by site-based alignment, keep the hypotheses in the percentage specified by this option. The valid range is 0 to 100	
	Valid values:	integers
	Default value:	<b>10</b>
	Minimum:	0
	Maximum:	100
<i>siteweight</i>	For calculating the Survival score, the weighting factor given for the site score. The valid range is 0.0 to 100.0	
	Valid values:	reals
	Default value:	<b>1</b>
	Minimum:	0.0
	Maximum:	100.0
<i>tolerance_a</i>		
	The feature matching tolerance for the hydrogen bond acceptor feature. The valid range is 0.0 to 100.0	

Valid values:    reals  
Default value: **1**  
Minimum:        0.0  
Maximum:        100.0

*tolerance\_d*

The feature matching tolerance for the hydrogen bond donor feature. The valid range is 0.0 to 100.0

Valid values:    reals  
Default value: **1**  
Minimum:        0.0  
Maximum:        100.0

*tolerance\_h*

The feature matching tolerance for the hydrophobic feature. The valid range is 0.0 to 100.0

Valid values:    reals  
Default value: **1.5**  
Minimum:        0.0  
Maximum:        100.0

*tolerance\_n*

The feature matching tolerance for the negative feature. The valid range is 0.0 to 100.0

Valid values:    reals  
Default value: **0.75**  
Minimum:        0.0  
Maximum:        100.0

*tolerance\_p*

The feature matching tolerance for the positive feature. The valid range is 0.0 to 100.0

Valid values:    reals  
Default value: **0.75**  
Minimum:        0.0  
Maximum:        100.0

*tolerance\_r*

The feature matching tolerance for the aromatic ring feature. The valid range is 0.0 to 100.0

Valid values:    reals  
Default value: **1.5**  
Minimum:        0.0  
Maximum:        100.0

*tolerance\_x*

The feature matching tolerance for the custom(X) feature. The valid range is 0.0 to 100.0

Valid values:    real  
Default value: **1**  
Minimum:    0.0  
Maximum:    100.0

*tolerance\_y*

The feature matching tolerance for the custom(Y) feature. The valid range is 0.0 to 100.0

Valid values:    real  
Default value: **1**  
Minimum:    0.0  
Maximum:    100.0

*tolerance\_z*

The feature matching tolerance for the custom(Z) feature. The valid range is 0.0 to 100.0

Valid values:    real  
Default value: **1**  
Minimum:    0.0  
Maximum:    100.0

*usetolerances*

When using feature matching tolerances, this option is true otherwise false.

Valid values:    boolean (true|false; yes|no; y|n; on|off)  
Default value: **false**

*vectorlowercutoff*

In case of score by vector-based alignment, keep the hypotheses that score above the value specified by this option. The valid range is 0.0 to 1.0

Valid values:    real  
Default value: **0.5**  
Minimum:    0.0  
Maximum:    1.0

*vectorweight*

For calculating the Survival score, the weighting factor given for the vector score. The valid range is 0.0 to 100.0

Valid values:    real  
Default value: **1**  
Minimum:    0.0  
Maximum:    100.0

*volumebasedalignment*

For score by volume-based alignment, this option is true otherwise false.

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **true**

*volumeweight*

For calculating the Survival score, the weighting factor given for the volume score. The valid range is 0.0 to 100.0

Valid values: reals  
Default value: **1**  
Minimum: 0.0  
Maximum: 100.0

## **phasescoreinactives**

Launches a Score Inactives job.

Syntax:

**phasescoreinactives** *matchscore=* $\langle x \rangle$

Options:

*matchscore*

For calculating the adjusted survival score, the weighting factor given for the inactive match score. The valid range is 0.0 to 100.0

Valid values: reals  
Default value: **1**  
Minimum: 0.0  
Maximum: 100.0

## **phasescoreselecthypothesis**

Selects the given hypothesis.

Syntax:

**phasescoreselecthypothesis** *alignall=yes | no*  $\langle ID \rangle$

Options:

**alignall** This option is used to control the behavior when filling in hypothesis alignments table. If this is “true” then alignments are generated for all ligands, including inactives, that match the hypothesis on at least 3 sites. If this is “false” then the table is filled in with only the ligands from the active set that match all of the hypothesis sites.  
Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **false**

Operands:

**⟨ID⟩**

The ID of the hypothesis to select. This will populate the Alignment table. This function is single select. If the operand is blank, the alignments table will be filled in for the currently selected hypothesis, if any.

## **phasesearchformatches**

Transfers the current hypothesis from Develop Common Pharmacophore panel (in either the Score Hypothesis or Build QSAR model steps) to the Find Matches panel. Also opens the Find Matches panel.

Syntax:

### **phasesearchformatches**

## **phaseselectalltablerows**

Selects all rows in the first table in the step.

Syntax:

### **phaseselectalltablerows**

## **phaseselectevrow**

Selects the given row in the excluded volumes table.

Syntax:

**phaseselectevrow** <row>

Operands:

<row>

The row number to select in the table.

**phaseselectextenddevrow**

Extends the selection to this row in the excluded volumes table.

Syntax:

**phaseselectextenddevrow** <row>

Operands:

<row>

The row number to extend the select to.

**phaseselectextendhypothesisrow**

Extends the selection to this row in the hypothesis table.

Syntax:

**phaseselectextendhypothesisrow** <row>

Operands:

<row>

The row number to extend the select to.

**phaseselectextendtablerow**

Extends the selection to this row in the table.

Syntax:

**phaseselectextendtablerow <row>**

Operands:

<row>

The row number to extend the select to.

**phaseselecthypothesisrow**

Selects the given row in the hypothesis table in the step.

Syntax:

**phaseselecthypothesisrow <row>**

Operands:

<row>

The row number to select in the hypothesis table.

**phaseselectonlyevrow**

Selects only this row in the excluded volumes table.

Syntax:

**phaseselectonlyevrow <row>**

Operands:

<row>

The row number to select only in the table row.

**phaseselectonlyhypothesisrow**

Selects only this row in the hypothesis table.

Syntax:

**phaseselectonlyhypothesisrow** <row>

Operands:

<row>

The row number to select only in the hypothesis table.

### **phaseselectonlytablerow**

Selects only this row in the table.

Syntax:

**phaseselectonlytablerow** <row>

Operands:

<row>

The row number to select only in the table row.

### **phaseselecttablerow**

Selects the given row in the first table in the step.

Syntax:

**phaseselecttablerow** <row>

Operands:

<row>

The row number to select in the table.

### **phasesetactivrows**

Toggles the Pharm Set property on or off for the given row in the ligands table in Prepare Ligands or Create Sites.

Syntax:

**phasesetactiverows** *value=active | inactive | none < rows >*

Options:

*value* The value of the Pharm set column in the ligands table in Prepare Ligands or Create Sites.  
Valid values: active  
inactive  
none  
Default value: **active**

Operands:

*< rows >*

The row numbers to toggle, two row numbers should be separated by , or if the user wants to specify a range then it can be given like 1-5 separated by a - . eg 1,2,5 or 1,4-7 .

## **phasesetactivity**

Sets the activity for the given row in the current step.

Syntax:

**phasesetactivity** *activity=< x > < row >*

Options:

*activity* The activity value for the given ligand.  
Valid values: reals  
Default value: **1**

Operands:

*< row >*

The row number of the ligand to set the activity for.

## **phasesetactivityproperty**

Sets the activity property.

Syntax:

**phasesetactivityproperty** <property>

Operands:

<property>

The property to get the activity values from.

## **phasesetactivitythresholds**

Sets the activity thresholds of the current run based on the current Phase options.

Syntax:

**phasesetactivitythresholds** *active\_threshold*=<text>  
                          *inactive\_threshold*=<text>

Options:

*active\_threshold*

The value used as the cutoff for assigning a ligand to the active Pharm Set. Can be left blank.

Valid values: text strings

Default value:

*inactive\_threshold*

The value used as the cutoff for assigning a ligand to the inactive Pharm Set. Can be left blank.

Valid values: text strings

Default value:

## **phasesetalignmentoptions**

Sets whether or not to view non-model ligands and whether or not to have a site mask.

Syntax:

**phasesetalignmentoptions** *alignnonmodel=yes | no*  
*sitemask=( text ) ( hypothesis )*

Options:

*alignnonmodel*

Indicates whether or not to align non-model ligands.

Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **false**

*sitemask* The site mask is a set of 0s and 1s indicating which sites to include in the mask. A 1 in a position means to include that site, and a 0 means to ignore it.

Valid values: text strings

Default value:

Operands:

*( hypothesis )*

The name of the hypothesis to set the options for.

## phasesetexcludedvolumes

Sets the value for the given cell.

Syntax:

**phasesetexcludedvolumes** *column=( n ) row=( n ) <value>*

Options:

*column* The column number of the cell to change.

Valid values: integers

Default value: **1**

*row* The row number of the cell to change.

Valid values: integers

Default value: **1**

Operands:

*<value>*

The value for the given cell.

## phasesetfrequency

Sets the minimum or maximum frequency for a given feature.

Syntax:

**phasesetfrequency** *column*=⟨n⟩ *value*=⟨n⟩ ⟨row⟩

Options:

*column* The table column to set.

Valid values: integers

Default value: **2**

Minimum: 1

*value* The minimum or maximum frequency.

Valid values: integers

Default value: **0**

Minimum: 0

Operands:

⟨row⟩

The row number of the min / max value to set.

## phasesettainingrows

Toggles the Training Set property on or off for the given row in the Alignments table under Build QSAR step.

Syntax:

**phasesettainingrows** *value*=training | test | none ⟨rows⟩

Options:

*value* The value of the training/test column in the alignment table under Build QSAR Model step.

Valid values: training

test

none

Default value: **training**

Operands:

$\langle \text{rows} \rangle$

The row numbers to toggle, two row numbers should be separated by , or if the user wants to specify a range then it can be given like 1-5 separated by a - . eg 1,2,5 or 1,4-7 .

## phasesiteoptions

Sets the site options for a Find Pharmacophores job.

Syntax:

**phasesiteoptions** *match*=all | minimum *minimum*= $\langle n \rangle$   
*numsites*= $\langle n \rangle$

Options:

*match*      Indicates whether to match against all or a minimum set of active ligands.

Valid values:    all  
                  minimum

Default value:    **all**

*minimum*      The minimum number of active ligands which a pattern has to match against.

Valid values:    integers  
Default value:    **2**  
Minimum:          2

*numsites*      The number of sites to match.

Valid values:    integers  
Default value:    **5**  
Minimum:          3  
Maximum:         7

## phasesorttable

Resort the given Phase table based on the data in the specified column

Syntax:

**phasesorttable** *table=⟨n⟩ ⟨column\_name⟩*

Options:

*table*      The table to set.

Valid values:      integers  
Default value:      **10**

Operands:

⟨column\_name⟩

The name of the column to be sorted.

## **phasestepforward**

Moves forward to the next Phase step. Deletes any steps after the current step, then creates the next step, using the data from previous steps.

Syntax:

**phasestepforward**

## **phasestepgoto**

Moves to an existing step in the current project.

Syntax:

**phasestepgoto** ⟨step name⟩

Operands:

⟨step name⟩

The name of the step to switch to.

## **phasetoggleactivetablerow**

Toggles the Active property on or off for the given row in the first table in the step.

Syntax:

**phasetoggleactivetablerow** <row>

Operands:

<row>

The row number to toggle.

## **phasetogglecentroidatom**

Adds or removes the given atom from the current centroid list for excluded volumes.

Syntax:

**phasetogglecentroidatom** <atom index>

Operands:

<atom index>

The index of the atom to toggle.

## **phasetoggletrainingrow**

Toggles the Training Set property on or off for the given row in the Build QSAR step.

Syntax:

**phasetoggletrainingrow** <row>

Operands:

<row>

The row number to toggle.

## **phaseundisplayproperty**

Undisplays the given property from the table.

Syntax:

**phaseundisplayproperty** < property name >

Operands:

< property name >

The property to undisplay.

## **phaseunmarkfeature**

Unmarks the given feature in the Workspace.

Syntax:

**phaseunmarkfeature** < feature >

Operands:

< feature >

A single letter (A-Z) indicating the feature to unmark.

## **phaseunselectevrow**

Unselects the given row in the excluded volumes table.

Syntax:

**phaseunselectevrow** < row >

Operands:

< row >

The row number to unselect in the table.

## **phaseunselecthypothesisrow**

Unselects the given row in the hypothesis table in the step.

Syntax:

**phaseunselecthypothesisrow** <row>

Operands:

<row>

The row number to unselect in the hypothesis table.

## **phaseunselecttablerow**

Unselects the given row in the first table in the step.

Syntax:

**phaseunselecttablerow** <row>

Operands:

<row>

The row number to unselect in the table.

## **picksize**

This command allows the user to choose the size of the pick box that is used for picking atoms/bonds/residues etc.

Syntax:

**picksize** <size>

Operands:

<size>

This operand actually defines the size of the pick box. Allowed sizes are 7X7, 10X10 and 15X15 which are specified at command line as 7, 10 & 15 respectively.

## **place**

Place the current fragment on screen at the <x>, <y> and <z> positions given by the operands.

Syntax:

**place** <x> <y> <z>

Operands:

<x> <y> <z>

The operands are three real numbers which are the x, y and z coordinates where the new fragment is to be placed.

## **plotxyarrangecolumn**

Puts all of the plots in a column.

Syntax:

**plotxyarrangecolumn**

## **plotxyarrangerow**

Displays the plots in a row.

Syntax:

**plotxyarrangerow**

## **plotxyarrangetiled**

Tiles all of the displayed plots.

Syntax:

## **plotxyarrangetiled**

### **plotxyaxis**

Creates or modifies an XY axis.

Syntax:

```
plotxyaxis axis=x | y maximum=<text> minimum=<text>  
      nummarkers=<n> plot=<text> <title>
```

Options:

*axis* This option indicates whether this axis is an X axis or a Y axis.

Valid values: **x**

**y**

Default value: **x**

*maximum* This option sets the maximum value for the axis.

Valid values: text strings

Default value:

*minimum* This option sets the minimum value for the axis.

Valid values: text strings

Default value:

*nummarkers*

This option sets the number of scale markers for the axis.

Valid values: integers

Default value: **10**

Minimum: 0

*plot* This option is the name of the plot containing the axis.

Valid values: text strings

Default value:

Operands:

*<title>*

The name of the axis to create or modify.

**plotxyaxisautorange**

Toggles autorange for the given axis on or off.

Syntax:

```
plotxyaxisautorange autorange=yes | no axis=x | y
plot=<text> <axisname>
```

Options:

*autorange* This option indicates whether or not to enable autorange for this axis.

Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **false**

*axis* This option indicates whether this axis is an X axis or a Y axis.

Valid values: **x**

**y**

Default value: **x**

*plot* This option is the name of the plot containing the axis.

Valid values: text strings

Default value:

Operands:

*<axisname>*

The name of the axis to set autorange for.

**plotxyaxisdelete**

Deletes an XY plot axis.

Syntax:

```
plotxyaxisdelete axis=x | y plot=<text> <axisname>
```

Options:

*axis* This option indicates whether this axis is an X axis or a Y axis.

Valid values: **x**

**y**

Default value: **x**

*plot*      This option is the name of the plot containing the axis.  
Valid values:    text strings  
Default value:

Operands:  
*<axisname>*  
The name of the axis to delete.

## **plotxyaxisdisplay**

Toggles display of the given axis on or off.

Syntax:

**plotxyaxisdisplay** *autorange=yes | no* *axis=x | y* *plot=<text>*  
*<axisname>*

Options:

*autorange*    This option indicates whether to display or undisplay the given axis.

Valid values:    boolean (true|false; yes|no; y|n; on|off)  
Default value:    **true**

*axis*      This option indicates whether this axis is an X axis or a Y axis.

Valid values:    x  
                  y  
Default value:    **x**

*plot*      This option is the name of the plot containing the axis.

Valid values:    text strings  
Default value:

Operands:

*<axisname>*  
The name of the axis to display or undisplay.

## **plotxyaxisrename**

Renames an existing axis.

Syntax:

**plotxyaxisrename** *axis=x | y newname=<text> plot=<text> <title>*

Options:

*axis* This option indicates whether this axis is an X axis or a Y axis.  
Valid values:    x  
                  y  
Default value:    x

*newname* This option is the new name for the axis.  
Valid values:    text strings  
Default value:

*plot* This option is the name of the plot containing the axis.  
Valid values:    text strings  
Default value:

Operands:

*<title>*  
The name of the axis to rename.

## **plotxycaption**

Sets the caption for the given plot.

Syntax:

**plotxycaption** *<plotname> <caption>*

Operands:

*<plotname> <caption>*  
The name of the plot to operate on. The new caption for the plot.

## **plotxycaptionposition**

Sets the position of the caption in the given plot.

Syntax:

**plotxycaptionposition** <plotname> top|bottom

Operands:

<plotname> top|bottom

The name of the plot to operate on. top to display the caption at the top of the plot or bottom to display it at the bottom.

## **plotxycopy**

Copies the selected plots.

Syntax:

**plotxycopy**

## **plotxydelete**

Deletes the selected plots.

Syntax:

**plotxydelete**

## **plotxydisplay**

Displays the selected plots in addition to any currently displayed plots.

Syntax:

**plotxydisplay**

## **plotxydisplaycaption**

Shows or hides the caption for the given plot.

Syntax:

**plotxydisplaycaption** <plotname> yes|no

Operands:

<plotname> yes|no

The name of the plot to operate on. yes to display the caption or no to not display the caption.

## **plotxydisplayincluded**

Sets the Display Included Markers state for the given plot.

Syntax:

**plotxydisplayincluded** <plotname> yes|no

Operands:

<plotname> yes|no

The name of the plot to operate on. yes to display included markers or no to not display them.

## **plotxydisplaylegend**

Displays or hides the legend for the given plot.

Syntax:

**plotxydisplaylegend** <plotname> yes|no

Operands:

<plotname> yes|no

The name of the plot to operate on. yes to display the legend or no to not display it.

## **plotxydisplayname**

Shows or hides the name in the caption for the given plot.

Syntax:

**plotxydisplayname** <plotname> yes|no

Operands:

<plotname> yes|no

The name of the plot to operate on. yes to display the plot name in the caption or no to not display the name in the caption.

## **plotxydisplayonly**

Displays only the selected plots.

Syntax:

**plotxydisplayonly**

## **plotxydisplaypointlabels**

Displays or hides the point labels for the given plot.

Syntax:

**plotxydisplaypointlabels** <plotname> yes|no

Operands:

<plotname> yes|no

The name of the plot to operate on. yes to display the point labels or no to not display them.

## **plotxydisplayselected**

Displays or hides the selected entry markers for the given plot.

Syntax:

**plotxydisplayselected** <plotname> yes|no

Operands:

<plotname> yes|no

The name of the plot to operate on. yes to display the selected entry markers or no to not display them.

## **plotxyhidesidebar**

Hides the plotxy side bar.

Syntax:

**plotxyhidesidebar**

## **plotxyhidetoolbar**

Hides the plotxy tool bar.

Syntax:

**plotxyhidetoolbar**

## **plotxylabel**

Turns on labels for the given data point.

Syntax:

**plotxylabel** *entryname*=yes | no *plot*=<text> *series*=<text>  
*xaxis*=yes | no *yaxis*=yes | no <entry>

Options:

*entryname*

If this is set to true then the entry name is displayed as part of the label.

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **true**

*plot*

This option is the name of the plot containing the series and entry.

Valid values: text strings  
Default value:

*series*

This option is the name of the series containing the entry.

Valid values: text strings  
Default value:

*xaxis*

If this is set to true then the X-axis property is displayed as part of the label.

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **true**

*yaxis*

If this is set to true then the Y-axis property is displayed as part of the label.

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **true**

Operands:

*<entry>*

The name of the entry to label.

## **plotxymove**

Moves the selected plots.

Syntax:

**plotxymove** *<location>*

Operands:

*<location>*

The location to move the selected plots to. If the location is 0, then the plots are moved to the beginning of the plot list. Otherwise, the plots are moved after the given plot number in the list.

## **plotxynew**

Creates a new XY plot

Syntax:

**plotxynew** <plotname>

Operands:

<plotname>

The name of the plot to create.

## **plotxypan**

Moves the selected plots.

Syntax:

**plotxypan** <x> <y>

Operands:

<x> <y>

This is a percentage of the full range to offset by. This is a percentage of the full range to offset by.

## **plotxypanplot**

Moves the specified plot.

Syntax:

**plotxypanplot** <plot> <x> <y>

Operands:

<plot> <x> <y>

This is the name of the plot to pan. This is a percentage of the full range to offset by. This is a percentage of the full range to offset by.

## **plotxyrename**

Renames the given plot to the new name.

Syntax:

**plotxyrename** <plotname> <newname>

Operands:

<plotname> <newname>

The name of the plot to operate on. The new name for the plot.

## **plotxyresetview**

Resets the zoom and pan for the selected plots.

Syntax:

**plotxyresetview**

## **plotxysaveimage**

Capture the current XY plotting window and save it to an image file.

Syntax:

**plotxysaveimage** *format=tiff | jpeg* <file\_name>

Options:

*format*      Specifies the format of the saved image.

Valid values:      tiff

                      jpeg

Default value:      tiff

Operands:

<file\_name>

The file where the image will be saved.

## **plotxyselect**

Selects the given plots.

Syntax:

**plotxyselect** <plotname>

Operands:

<plotname>

Names of the plots to select, or all, or displayed.

## **plotxyseries**

Creates or modifies an XY data series.

Syntax:

```
plotxyseries color=black | red | green | blue | purple | orange |
    blue_green | light_green | red_purple | yellow | cyan
    marker=filled_square | square | filled_circle | circle | cross |
    point | diamond | filled_diamond | none plot=<text>
    style=none | solid | dash unityaspect=yes | no width=<n>
    xaxis=<text> xbuckets=<n> xproperty=<text> yaxis=<text>
    ybuckets=<n> yproperty=<text> <seriesname>
```

Options:

*color*      The color of the symbols and lines used on the plot for this data series.

Valid values:	black red green blue purple orange blue_green light_green red_purple yellow cyan
---------------	--

Default value: **black**

<i>marker</i>	The marker for this data series. Valid values: filled_square square filled_circle circle cross point diamond filled_diamond none Default value: <b>filled_square</b>
<i>plot</i>	This option is the name of the plot containing the series. Valid values: text strings Default value:
<i>style</i>	The line style for this data series. Valid values: none solid dash Default value: <b>none</b>
<i>unityaspect</i>	Settings this option to yes causes the plot to display with an equal increment on each axis mapping to the same distance on the screen. Valid values: boolean (true false; yes no; y n; on off) Default value: <b>false</b>
<i>width</i>	This option sets the width of the line for this data series. Valid values: integers Default value: <b>1</b> Minimum: 1
<i>xaxis</i>	This option selects an existing X-axis. Valid values: text strings Default value:
<i>xbuckets</i>	This option sets the number of buckets for the X-axis. Valid values: integers Default value: <b>100</b> Minimum: 0
<i>xproperty</i>	This option sets the name of the property for the X-axis. Valid values: text strings Default value:

<i>yaxis</i>	This option selects an existing Y-axis. Valid values: text strings Default value:
<i>ybuckets</i>	This option sets the number of buckets for the Y-axis. Valid values: integers Default value: <b>100</b> Minimum: 0
<i>yproperty</i>	This option sets the name of the property for the Y-axis. Valid values: text strings Default value:

Operands:

*<seriesname>*

The name of the series to create or modify.

## **plotxyseriesdelete**

Deletes the given XY data series.

Syntax:

**plotxyseriesdelete** *plot=<text> <seriesname>*

Options:

<i>plot</i>	This option is the name of the plot containing the series. Valid values: text strings Default value:
-------------	--

Operands:

*<seriesname>*

The name of the series to delete.

## **plotxyseriesdisplay**

Displays or undisplays the given series.

Syntax:

**plotxyseriesdisplay** <plotname> <series> yes|no

Operands:

<plotname> <series> yes|no

The name of the plot to operate on. The name of the series. yes to display the given data series or no to not display it.

## **plotxyseriesrename**

Renames an existing series.

Syntax:

**plotxyseriesrename** newname=<text> plot=<text> <title>

Options:

*newname* This option is the new name for the series.

Valid values: text strings

Default value:

*plot* This option is the name of the plot containing the series.

Valid values: text strings

Default value:

Operands:

<title>

The name of the series to rename.

## **plotxyseriesselect**

Selects only the entries corresponding to the given series.

Syntax:

**plotxyseriesselect** *plot=⟨text⟩ ⟨seriesname⟩*

Options:

*plot* This option is the name of the plot containing the series.

Valid values: text strings

Default value:

Operands:

⟨seriesname⟩

The name of the series to select.

**plotxyseriesselectadd**

Adds the entries corresponding to the given series to the current selection.

Syntax:

**plotxyseriesselectadd** *plot=⟨text⟩ ⟨seriesname⟩*

Options:

*plot* This option is the name of the plot containing the series.

Valid values: text strings

Default value:

Operands:

⟨seriesname⟩

The name of the series to select.

**plotxyshowsidebar**

Displays the plotxy side bar.

Syntax:

**plotxyshowsidebar**

**plotxyshowtoolbar**

Displays the plotxy tool bar.

Syntax:

**plotxyshowtoolbar**

**plotxytoggledisplay**

Toggles the display of the plot on or off.

Syntax:

**plotxytoggledisplay** <plotname>

Operands:

<plotname>

The name of the plot to toggle the display of.

**plotxyundisplay**

Undisplays the selected plots.

Syntax:

**plotxyundisplay**

**plotxyunlabel**

Turns off the label for the given data point.

Syntax:

**plotxyunlabel** *plot*=⟨text⟩ *series*=⟨text⟩ ⟨entry⟩

Options:

*plot*      This option is the name of the plot containing the series and entry.

Valid values:    text strings

Default value:

*series*    This option is the name of the series containing the entry.

Valid values:    text strings

Default value:

Operands:

⟨entry⟩

The name of the entry to unlabel.

## **plotxyunselect**

Unselects the given plots.

Syntax:

**plotxyunselect** ⟨plotname⟩

Operands:

⟨plotname⟩

Names of the plots to unselect, or all, or displayed.

## **plotxyupdate**

Updates the selected plots.

Syntax:

## **plotxyupdate**

### **plotxyzoom**

Scales the selected plots.

Syntax:

**plotxyzoom** <x> <y>

Operands:

<x> <y>

This is the amount of scaling to use in the horizontal direction. This is the amount of scaling to use in the vertical direction.

### **plotxyzoompan**

Scales and offsets the given plot.

Syntax:

**plotxyzoompan** <plot> <zoomx> <zoomy> <panx> <pany>

Operands:

<plot> <zoomx> <zoomy> <panx> <pany>

This is the name of the plot to apply the zoom and pan factors to. This is the amount of scaling to use in the horizontal direction. This is the amount of scaling to use in the vertical direction. This is the amount to offset in the horizontal direction as a percentage of the full range. This is the amount to offset in the vertical direction as a percentage of the full range.

## **posereceptor**

Define attribute of pose receptor in the pose viewer.

Syntax:

**posereceptor** *display=yes | no*

Options:

*display* If this option is set to “false” then the pose file receptor will not be displayed in the 3D workspace.  
 Valid values: boolean (true|false; yes|no; y|n; on|off)  
 Default value: **true**

**potential**

Set various options associated with the definition of the potential energy to be used in a MacroModel job.

Syntax:

**potential** *cele=<x> charges=force\_field | structure\_file chnd=<x> cutoff=normal | extended | user\_defined | none cvdw=<x> dielectric=<x> electrostatics=field\_field | constant | distance\_dependant field=mm2\* | mm3\* | amber\* | opls\* | amber94 | mmff | mmffs | oplsaa | opls2005 | opls2007 solvent=none | water | chcl3 | octanol substructure=yes | no*

Options:

*cele* This option determines what cutoff will be used for the electrostatic part of the energy calculation.  
 Valid values: reals  
 Default value: **12**  
 Minimum: 0.0  
 Maximum: 99999.0

*charges* This option determines where the charges to be used in the energy calculation will come from.  
 Valid values: force\_field  
                   structure\_file  
 Default value: **force\_field**

*chnd* This option determines what cutoff will be used for the hydrogen bond part of the energy calculation.  
 Valid values: reals  
 Default value: **4**  
 Minimum: 0.0  
 Maximum: 99999.0

<i>cutoff</i>	This option determines what type of non-bonded cutoff will be used in the energy calculation.
	Valid values:    normal extended user_defined none
	Default value: <b>normal</b>
<i>cvdw</i>	This option determines what VDW cutoff will be used in the energy calculation.
	Valid values:    reals
	Default value: <b>7</b>
	Minimum:        0.0
	Maximum:        99999.0
<i>dielectric</i>	The dielectric constant to be used in the electrostatic part of the energy calculation.
	Valid values:    reals
	Default value: <b>1</b>
	Minimum:        0.999999999
<i>electrostatics</i>	The electrostatic treatment to be used in the energy calculation.
	Valid values:    field_field constant distance-dependant
	Default value: <b>constant</b>
<i>field</i>	The force field to be used for the energy calculation.
	Valid values:    mm2* mm3* amber* opls* amber94 mmff mmffs oplsaa opls2005 opls2007
	Default value: <b>opls2005</b>
<i>solvent</i>	The solvent model to be used for the energy calculation
	Valid values:    none water chcl3 octanol
	Default value: <b>none</b>

*substructure*

[NOTE: This option is no longer used.]

Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **false**

**pprep**

This keyword is used to set various options associated with running protein preparation.

Syntax:

```
pprep cavity=default | liaison | no_neutralization procedure=both |  
prepare | refine rmsd=⟨x⟩
```

Options:

<i>cavity</i>	Which preparation procedure to run. Both preparation and refinement by default.
	Valid values: default liaison no_neutralization
	Default value: <b>no_neutralization</b>
<i>procedure</i>	Which preparation procedure to run. Both preparation and refinement by default.
	Valid values: both prepare refine
	Default value: <b>both</b>
<i>rmsd</i>	Minimum RMSD.
	Valid values: reals
	Default value: <b>0.3</b>
	Minimum: 0.0

**ppreceptorligand**

Defines a on-screen molecule to be treated as the ligand for a Pprep calculation

Syntax:

**ppreceptorligand** <molecule\_number>

Operands:

<molecule\_number>

The number of a molecule to be included as the ligand.

## **pprepwrite**

Write the files required for protein structure preparation.

Syntax:

## **pprepwrite**

## **prefer**

Set the global preferences for handling markers (derived graphical objects). Specifically, specify under which conditions markers will be deleted.

Syntax:

```
prefer autofit=never | singleentry | always beep=yes | no
buildbackbonesubjobs=<n> collapsed=<n>
commandinputshow=yes | no deletemarkers=mismatch |
missing displayeditwarning=yes | no feedbackproperty=<text>
feedbackshow=yes | no initworkdir=<text>
keepcombiglidejobfiles=yes | no keepphasejobfiles=yes | no
lastphasedb=<text> lastproject=<text>
macromodelsuffix=<text> maestrosuffix=<text>
mol2suffix=<text> openlastproject=yes | no pdbsuffix=<text>
phasetolerance_a=<x> phasetolerance_d=<x>
phasetolerance_h=<x> phasetolerance_n=<x>
phasetolerance_p=<x> phasetolerance_r=<x>
phasetolerance_x=<x> phasetolerance_y=<x>
phasetolerance_z=<x> projectsave=small | medium
projectsuffix=<text> projectsync=auto | prompt | manual
rightclicktimerperiod=<n> sdsuffix=<text>
sequenceviewerproximity=<x> sequenceviewershow=yes | no
showentryname=yes | no showprojecttable=yes | no
statusbarshow=yes | no tbcollapsed=<n> tblocation=<n>
toolbarshow=yes | no topviewshow=yes | no
workspacemenushow=yes | no
```

Options:

*autofit* Specifies whether workspace entries fit to screen automatically or not. Valid values are “never”, “singleentry”, and “always”. These cause Maestro never fit to screen, fit when only one entry in workspace and always fit to screen respectively.

Valid values:    **never**  
                   **singleentry**  
                   **always**

Default value: **never**

*beep* Whether or not the system beeps for picking feedback.

Valid values:    boolean (true|false; yes|no; y|n; on|off)

Default value: **true**

*buildbackbonesubjobs*

This is the number of templates to run at one time in Build Backbone.

Valid values:    integers  
                   **1**  
                   Minimum:    **1**

*collapsed* Whether the status bar is collapsed or not.

Valid values:    integers  
                   Default value: **0**

*commandinputshow*

Whether the command input area is show or hide.

Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **false**

*deletemarkers*

This option sets the condition for marker deletion. Valid values are “mismatch” or “missing”.

Valid values: mismatch

missing

Default value: **mismatch**

*displayeditwarning*

If set to true, then a warning message is displayed when user edits value of an entry property in PT.

Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **true**

*feedbackproperty*

This is the property displayed at the end of the feedback string in workspace.

Valid values: text strings

Default value: **Title**

*feedbackshow*

Whether the workspace feedback is activated.

Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **true**

*initworkdir*

The Maestro input/output directory is used for starting jobs and for other file input and output. The i/o directory is always displayed in the title bar of the Main Application Window. The i/o dir can be set to a number of values which affect Maestro’s behavior when reading and writing files and when running jobs. You can specify “currentdir” (“startdir” is a synonym) to set the i/o directory to use Maestro’s current working directory (cwd), “project” to change to the project directory, “projectparent” to change to the parent directory which contains the project directory, or “projectjobs” to change to the “jobs” directory within the project directory. Otherwise, specify a directory path to be used. Maestro has a current working directory (cwd), similar to what a Unix shell has. When “currentdir” is in effect job files will be placed in the cwd. This is located at the very top of the window next to the window’s border. The cwd is changed whenever a cd (changedirectory) command

Valid values: text strings  
Default value: **currentdir**

*keepcombiglidejobfiles*

If this is set to true, then CombiGlide job files will be preserved after a job completes. Otherwise, the job files will be removed when each job completes.

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **true**

*keepphasejobfiles*

If this is set to true, then Phase job files will be preserved after a job completes. Otherwise, the job files will be removed when each job completes.

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **false**

*lastphasedb*

This records the last Phase 3D Database which was opened.

Valid values: text strings  
Default value:

*lastproject* This records the last project which was opened.

Valid values: text strings  
Default value:

*macromodelsuffix*

The default suffix or extension to be used for exported Macro-Model files. The suffix will automatically be appended to a file name if it has no suffix.

Valid values: text strings  
Default value: **dat**

*maestrosuffix*

The default suffix or extension to be used for exported Maestro files. The suffix will automatically be appended to a file name if it has no suffix.

Valid values: text strings  
Default value: **mae**

*mol2suffix*

The default suffix or extension to be used for exported Mol2 files. The suffix will automatically be appended to a file name if it has no suffix.

Valid values: text strings  
Default value: **mol2**

*openlastproject*

If set to true, then when Maestro starts it will re-open the last project that was open when Maestro was closed.

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **false**

*pdbsuffix* The default suffix or extension to be used for exported PDB files. The suffix will automatically be appended to a file name if it has no suffix.

Valid values: text strings  
Default value: **pdb**

*phasetolerance\_a*

The feature matching tolerance for the hydrogen bond acceptor feature. The valid range is 0.0 to 100.0

Valid values: reals  
Default value: **1**  
Minimum: 0.0  
Maximum: 100.0

*phasetolerance\_d*

The feature matching tolerance for the hydrogen bond donor feature. The valid range is 0.0 to 100.0

Valid values: reals  
Default value: **1**  
Minimum: 0.0  
Maximum: 100.0

*phasetolerance\_h*

The feature matching tolerance for the hydrophobic feature. The valid range is 0.0 to 100.0

Valid values: reals  
Default value: **1.5**  
Minimum: 0.0  
Maximum: 100.0

*phasetolerance\_n*

The feature matching tolerance for the negative feature. The valid range is 0.0 to 100.0

Valid values: reals  
Default value: **0.75**  
Minimum: 0.0  
Maximum: 100.0

*phasetolerance\_p*

The feature matching tolerance for the positive feature. The valid range is 0.0 to 100.0

Valid values: reals  
Default value: **0.75**  
Minimum: 0.0

Maximum: 100.0

*phasetolerance\_r*

The feature matching tolerance for the aromatic ring feature.  
The valid range is 0.0 to 100.0

Valid values: reals

Default value: **1.5**

Minimum: 0.0

Maximum: 100.0

*phasetolerance\_x*

The feature matching tolerance for the custom(X) feature. The  
valid range is 0.0 to 100.0

Valid values: reals

Default value: **1**

Minimum: 0.0

Maximum: 100.0

*phasetolerance\_y*

The feature matching tolerance for the custom(Y) feature. The  
valid range is 0.0 to 100.0

Valid values: reals

Default value: **1**

Minimum: 0.0

Maximum: 100.0

*phasetolerance\_z*

The feature matching tolerance for the custom(Z) feature. The  
valid range is 0.0 to 100.0

Valid values: reals

Default value: **1**

Minimum: 0.0

Maximum: 100.0

*projectsolve*

Specifies information to be saved to disk when project is closed.  
The “small” option saves only the compressed opening state.  
Because it takes longer to open and close projects when the  
current state must be expanded or deleted, this option is not  
recommended unless you are low on disk space. The “medium”  
option saves the compressed opening state and the expanded  
current project state. This option requires more disk space, but  
provides faster and easier access and the safeguard of redundant  
project data. It also allows access to the project by 4.1 and  
5.0 versions of Maestro that cannot use the compressed opening  
state.

Valid values: small  
medium  
Default value: **medium**

*projectsuffix*

The default suffix or extension to be used for project directories. When a project selector is used in Maestro to choose a project, the filter is automatically set to match this suffix. The suffix is automatically appended, if missing, to the returned project path when there is potential for creating a new project (projectnew, projectrename, or projectcopy). However, when a project command is issued without using a project selector (e.g. in a macro, script, or the command input area), the suffix is not automatically appended. Also, in cases where an existing project must be opened (projectopen, projectmerge) the suffix is not appended.

Valid values: text strings  
Default value: **.prj**

*projectsync*

Specifies whether workspace changes to project entries are automatically saved or not. Valid values are “auto”, “prompt”, and “manual”. These cause Maestro to save changes automatically, prompt to save changes, or save changes only when explicitly directed by the user.

Valid values: auto  
prompt  
manual  
Default value: **auto**

*rightclicktimerperiod*

The time delay for workspace menu to appear in milliseconds.

Valid values: integers  
Default value: **300**

*sdsuffix*

The default suffix or extension to be used for exported SD files. The suffix will automatically be appended to a file name if it has no suffix.

Valid values: text strings  
Default value: **sdf**

*sequenceviewerproximity*

Cutoff distance for proximity coloring in the sequence viewer

Valid values: reals  
Default value: **4**  
Minimum: 0.0

*sequenceviewershows*

Whether the sequence viewer is show or hide.

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **false**

*showentryname*

If set to true, then Entry Name property column is shown in the PT otherwise its hidden by default.

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **false**

*showprojecttable*

If set to true, then when a project is opened, the project table will automatically be displayed.

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **true**

*statusbarshow*

Whether the status bar is show or hide.

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **true**

*tbcollapsed*

Whether the tool bar is collapsed or not.

Valid values: integers  
Default value: **0**

*tblocation*

Whether the tool bar is located left or right.

Valid values: integers  
Default value: **1**

*toolbarshow*

Whether the tool bar is show or hide.

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **true**

*topviewshow*

Whether the top view window is show or hide.

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **false**

*workspacemenushow*

Whether the workspace menu should be displayed on right click.

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **true**

## **previouspose**

If there is more than one pose structure in the pose viewer, display the one before the first displayed pose. Upon reaching the beginning of the list, wraps around to the end.

Syntax:

## **previouspose**

## **print**

Print to a postscript printer or file. The print options must be set independently using the “printoptions” command. If an operand is given then the postscript is written to a file of that name, otherwise it is sent directly to the printer.

Syntax:

**print** [*<file\_name>*]

Operands:

[*<file\_name>*]

If an operand is given it is treated as the name of the file to which the postscript will be written. The full filename including any suffix should be specified.

## **printoptions**

Set up options for the Print command, which sends the on-screen images to a postscript printer or file. If the destination is an encapsulated postscript file (eps), the command, orientation, scale\_to\_page, and title options are ignored

Syntax:

**printoptions** *color\_mode=color | black\_text | black\_text\_and\_lines*  
    *command=<text> file\_format=ps | eps orientation=portrait |*  
    *landscape print\_background=yes | no scale\_to\_page=yes | no*  
    *smoothness=<n> title=<text>*

Options:

*color\_mode*

Specifies one of three modes: 1) full color (grey-scale on a black and white printer); 2) black text; and 3) black text and lines.

Valid values:    **color**  
                  **black\_text**  
                  **black\_text\_and\_lines**

Default value:    **color**

*command*    This is the command used to print to a postscript printer.

Valid values:    text strings  
Default value:    **lpr**

*file\_format*

When printing to a file, specifies whether to generate postscript (ps) or encapsulated postscript (eps).

Valid values:    **ps**  
                  **eps**  
Default value:    **ps**

*orientation*

Specifies portrait or landscape orientation (ignored in eps files).

Valid values:    **portrait**  
                  **landscape**  
Default value:    **portrait**

*print\_background*

Specifies whether or not to print the background color.

Valid values:    boolean (true|false; yes|no; y|n; on|off)  
Default value:    **false**

*scale\_to\_page*

Specifies whether or not to scale the screen image to fit page (ignored in eps files).

Valid values:    boolean (true|false; yes|no; y|n; on|off)  
Default value:    **false**

*smoothness*

Determines how well smoothly shaded surfaces are rendered. The highest value (10) will result in the best image, however, the lowest value (1) will result in a shorter print file and faster printing.

	Valid values:	integers
	Default value:	<b>6</b>
	Minimum:	1
	Maximum:	10
<i>title</i>	Optional title text, which is added to the bottom of the page (ignored in eps files).	
	Valid values:	text strings
	Default value:	

## **projectclose**

Close the current project and open a new scratch project.

Syntax:

## **projectclose**

## **projectcopy**

Make a copy of the current project.

Syntax:

```
projectcopy entry=yes | no job=yes | no plot=yes | no  
run=yes | no snapshot=yes | no table=yes | no  
user=yes | no <to_dir_path>
```

Options:

<i>entry</i>	Enable/disable copying of project entry files. Valid values: boolean (true false; yes no; y n; on off) Default value: <b>true</b>
<i>job</i>	Enable/disable copying of project job files. Valid values: boolean (true false; yes no; y n; on off) Default value: <b>false</b>
<i>plot</i>	Enable/disable copying of project plots. If this option is enabled, entries should also be copied. Valid values: boolean (true false; yes no; y n; on off) Default value: <b>true</b>

<i>run</i>	Enable/disable copying of project run files. Valid values: boolean (true false; yes no; y n; on off) Default value: <b>true</b>
<i>snapshot</i>	Enable/disable copying of project snapshot. Valid values: boolean (true false; yes no; y n; on off) Default value: <b>true</b>
<i>table</i>	Enable/disable copying of project table files. Valid values: boolean (true false; yes no; y n; on off) Default value: <b>true</b>
<i>user</i>	Enable/disable copying of project user files. Valid values: boolean (true false; yes no; y n; on off) Default value: <b>false</b>

Operands:

*<to\_dir\_path>*

The path (name and location) of the directory to be used to copy the current project. This can either be a new or existing project directory.

## **projectdelete**

Delete the current project and open a new scratch project.

Syntax:

## **projectdelete**

## **projectmerge**

Merge data from another project into the current Maestro project.

Syntax:

**projectmerge** *entry=yes | no job=yes | no run=yes | no user=yes | no <from\_dir\_path>*

Options:

<i>entry</i>	Enable/disable merging of project entry files. Valid values: boolean (true false; yes no; y n; on off) Default value: <b>true</b>
<i>job</i>	Enable/disable merging of project job files. Valid values: boolean (true false; yes no; y n; on off) Default value: <b>false</b>
<i>run</i>	Enable/disable merging of project run files. Valid values: boolean (true false; yes no; y n; on off) Default value: <b>true</b>
<i>user</i>	Enable/disable merging of project user files. Valid values: boolean (true false; yes no; y n; on off) Default value: <b>false</b>

Operands:

*<from\_dir\_path>*

The path (name and location) of the project directory to be merged into the current project.

## **projectnew**

Create a new project and open it in Maestro.

Syntax:

**projectnew <dir\_path>**

Operands:

*<dir\_path>*

The path (name and location) of the directory to be created as the new project directory.

## **projectopen**

Open an existing project into Maestro.

Syntax:

**projectopen** <dir\_path>

Operands:

<dir\_path>

The path (name and location) of the project directory to be opened.

**projectrename**

Rename the current project directory and/or move it to a new location.

Syntax:

**projectrename** <to\_dir\_path>

Operands:

<to\_dir\_path>

The new path (name and location) for the current project directory.

**projectrevertopen**

Revert state of current project to that it had when opened in Maestro.

Syntax:

**projectrevertopen**

**projectrevertsnapshot**

Revert state of current project to that previously stored in snapshot.

Syntax:

## **projectrevertsnapshot**

### **projectsaveas**

Save project and place user into that project

Syntax:

**projectsaveas** <to\_dir\_path>

Operands:

<to\_dir\_path>

Name to which the project will be saved. Saves all data from the current project (table, plots, etc.)

## **projectstoresnapshot**

Save copy of current project state for reversion.

Syntax:

**projectstoresnapshot**

## **projectsynchronize**

Save changes in workspace to current project.

Syntax:

**projectsynchronize**

## **projecttablefind**

Searches for a string in the project table and make that cell editable.

Syntax:

```
projecttablefind direction=up | down matchcase=yes | no
matchword=yes | no searchhiddenrows=yes | no <find_string>
```

Options:

*direction* With this option user can specify the direction of search.

Valid values: up  
down

Default value: **down**

*matchcase* With this option user can specify for case sensitive search.

Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **false**

*matchword*

With this option user can specify for matching word as a whole.

Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **false**

*searchhiddenrows*

With this option user can specify whether to search in collapsed group rows or not.

Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **false**

Operands:

*<find\_string>*

Its the string for which the user want to search in the project table.

## **projectupdatecoordinates**

Update the coordinates for the included entries in the project. This applies any current transformations in the Workspace to the original coordinates and places the result back into the project, overwriting the original coordinates.

Syntax:

## **projectupdatecoordinates**

## **propertycalculate**

This command calculates the given property for the entries that are specified with the ESL.

Syntax:

**propertycalculate** *propertynamne=(text)* *recalculate=yes | no*  
                  *(ESL)*

Options:

*propertynamne*

This option is the name of the property that need to be calculated. The valid values for this option are numatoms , numresidues , nummolecules , molweight , spin , molcharge and secstruct for calculating the properties Number of atoms , Number of residues , Number of molecules , Molecular weight , Spin multiplicity , Molecular charge and Secondary Structure Content respectively. A new property will be added to the project this <propertynamne> and the values will be assigned to those entries which are match the ESL.

Valid values: text strings

Default value: **numatoms**

*recalculate* The valid values are true and false. With this option the user can specify whether to re-calculate the property data, which has been calculated already. This option will not have any impact on the entries which does not have the property data. If this option value is true, then the property value will be calculated and the new value will be assigned. If the options value is false, then the already existing property values will not be calculated.

Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **false**

Operands:

*(ESL)*

A valid ESL specification. Calculates specified property for those entries which match the ESL description.

**propertycalculatemolformula**

This command calculates the molecular formula for the entries that are specified with the ESL.

Syntax:

**propertycalculatemolformula** *maxatoms=⟨n⟩*  
*recalculate=yes | no ⟨ESL⟩*

Options:

*maxatoms* This option is used for calculating molecular formula for an entry which has only one molecule. If the entry contains more than the specified maxatom then the molecular formula will not be calculated for that entry.

Valid values: integers

Default value: **100**

*recalculate* The valid values are true and false. With this option the user can specify whether to re-calculate the property data, which has been calculated already. This option will not have any impact on the entries which does not have the property data. If this option value is true, then the property value will be calculated and the new value will be assigned. If the options value is false, then the already existing property values will not be calculated.

Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **false**

Operands:

⟨ESL⟩

A valid ESL specification. Calculates molecular formula for those entries which match the ESL description.

**propertycalculatepickpka**

This command calculates the pickPka atom for the entries that are specified with the ESL.

Syntax:

**propertycalculatepickpka** *atomname*=⟨text⟩  
    *recalculate*=yes | no ⟨ESL⟩

Options:

*atomname* This option is the name of the atom that is to be set as the pKa atom for the Jaguar calculation.

Valid values: text strings

Default value:

*recalculate* The valid values are true and false. With this option the user can specify whether to re-calculate the property data, which has been calculated already. This option will not have any impact on the entries which does not have the property data. If this option value is true, then the property value will be calculated and the new value will be assigned. If the options value is false, then the already existing property values will not be calculated.

Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **false**

Operands:

⟨ESL⟩

A valid ESL specification. Selects the pickPka atom for those entries which match the ESL description.

## **propertycalculatesubstructs**

This command calculates the number of sub-structures for the entries that are specified with the ESL.

Syntax:

**propertycalculatesubstructs** *name*=⟨text⟩  
    *recalculate*=yes | no *substructure*=⟨text⟩ ⟨ESL⟩

Options:

*name* This option holds the name of substructure. A valid property name is the valid value for this option. A new property of the integer type will be created with this <name>, which holds the count of specified sub-structures in an entry.

Valid values: text strings

Default value:

*recalculate* The valid values are true and false. With this option the user can specify whether to re-calculate the property data, which has been calculated already. This option will not have any impact on the entries which does not have the property data. If this option value is true, then the property value will be calculated and the new value will be assigned. If the options value is false, then the already existing property values will not be calculated.

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **false**

*substructure*

This describes the sub-structure defintion. This value is used for matching substructures in each entry. The count of these sub-structures in each entry is saved as a property in the table.

Valid values: text strings  
Default value:

Operands:

$\langle \text{ESL} \rangle$

A valid ESL specification. Calculates the number of sub-structures for those entries which match the ESL description.

## **propertyclearvalue**

This command clear values of the given properties

Syntax:

**propertyclearvalue** *allentries=false | true*  $\langle \text{propertynames} \rangle$

Options:

*allentries* The property value to be cleared for selected entries or all the entries. This has two valid values, true for clearing all property values and false for clearing property values for the selected entries only.

Valid values: **false**  
**true**  
Default value: **false**

Operands:

$\langle \text{propertynames} \rangle$

The names of properties for which values need to be cleared.

## **propertycreate**

Create a new property for entries in the current project without assigning any entry property values.

Syntax:

```
propertycreate author=⟨text⟩ displayprecision=⟨n⟩ type=bool |  
int | double | string ⟨property_name⟩
```

Options:

*author*      The m2io signature of the owner (e.g. person, group, or software module), or authority, that defines the property (meaning and range of values) and generally assigns its values. The author for user-defined properties should normally be user , as these properties can be freely edited in the project table. Be careful if setting author to m , mmod , i , j , qp , sd , or other names reserved for properties generated by existing programs.

Valid values:    text strings

Default value:    **user**

*displayprecision*

The Display precision for the real data type variable for display use only.

Valid values:    integers

Default value:    **4**

Minimum:        0

Maximum:        15

*type*

The manner in which the property values are to be stored and represented, either bool (for Boolean), int (for integer), double (for double precision floating point, real numbers), or string (for text character strings).

Valid values:    bool

                  int

                  double

                  string

Default value:    **int**

Operands:

`<property_name>`

The user-assigned name of the property, which `propertyrename` can alter. The property name must be unique within the current project. If it is, a unique m2io data name will be generated by combining the type, author, and name, possibly modified (such as replacing spaces with underscores and adding a number at the end) to make it valid and unique.

## **propertydelete**

This command deletes the given properties.

Syntax:

**propertydelete** `<propertynames>`

Operands:

`<propertynames>`

The names of properties to delete.

Aliases:

**deleteproperty** (see [\[deleteproperty\]](#), page 87)

## **propertygeneratecontacts**

This command applies a Contacts measurement as a property to a number of selected entries in a project. It is valid at least `Contactset1` has been set. If only `Contactset1` is defined, then `Contactset2 = Contactset1`. Normally these two sets are required having been set before this command, otherwise no property will be generated.

Syntax:

### **propertygeneratecontacts**

### **propertygeneratehbond**

This command applies an HBond measurement as a property to a number of selected entries in a project. It is valid at least HBondset1 has been set. If onle HBondset1 is defined, then HBondset2 = HBondset1. Normally these two sets are required having been set before this command, otherwise no property will be generated.

Syntax:

### **propertygeneratehbond**

### **propertymeasurementsetting**

This command sets whether a measurement is applied as a property to a number of selected entries in a project.

Syntax:

### **propertymeasurementsetting** *applytoselectedentries=yes | no*

Options:

#### *applytoselectedentries*

This option determines whether to apply a measurement as a property to a number of selected entries in a project.

Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **false**

### **propertymove**

Moves the property in the first column to the second column.

Syntax:

### **propertymove** <from> <to> <table>

Operands:

<from> <to> <table>

The source column number. The destination column number. The name of the table to use as the source for the column numbers. If the table operand is missing, the current or default table (1) will be used.

### **propertyprecision**

This command sets the precision of the property.

Syntax:

### **propertyprecision** <property\_name> <precision>

Operands:

<property\_name> <precision>

The name of the property. Precision for that property.

### **propertyrename**

This command renames the given property to the given name.

Syntax:

### **propertyrename** <from> <to>

Operands:

<from> <to>

The name of the property to rename. The name to rename the property to.

Aliases:

[renameproperty](#) (see [renameproperty], page 414)

## **propertyshowall**

This command creates a property subset consisting of all the properties in the project.

Syntax:

### **propertyshowall**

## **propertysuperimposesetting**

This command sets whether a superimposition is applied as a property to any entry in the project; and if applied, to which entries it is applied i.e. to the selected entries or to the included entries in the project.

Syntax:

**propertysuperimposesetting** *applytoentries*=included | selected  
                          *applytoincludedentries*=yes | no   *createproperty*=yes | no

Options:

### *applytoentries*

This option determines to which entries the superimposition is applied as a property. Valid values are “selected”, or “included”.

Valid values:    included  
                      selected

Default value:    **included**

### *applytoincludedentries*

This option determines whether to apply a superimposition as a property to a number of included entries in a project. NOTE: This option is deprecated now onwards and is supposed just for backward compatibility. Internally the option will be interpreted as: true => applytoentries = included createproperty = true false => createproperty = false

Valid values:    boolean (true|false; yes|no; y|n; on|off)  
Default value:    **false**

### *createproperty*

Valid values:    boolean (true|false; yes|no; y|n; on|off)  
Default value:    **false**

## protassign

This keyword is used to set various options associated with running protassign jobs.

Syntax:

```
protassign input_file=⟨text⟩ structure_source=selected_entries |  
    workspace | file
```

Options:

*input\_file* The name of the structure input file.

Valid values: text strings

Default value:

*structure\_source*

Whether to use the selected entries in the current project, or what is in the workspace, or a specified file with multiple structures as structure input for the job.

Valid values: selected\_entries

workspace

file

Default value: **workspace**

## protassignresidues

Defines a set of atoms for protein assignment.

Syntax:

```
protassignresidues ⟨ASL⟩
```

Operands:

⟨ASL⟩

The ASL expression which defines the atoms that will be used to define the residues for protein assignment.

## protassignstart

Start a protein assignment job with the current settings.

Syntax:

**protassignstart**

**protassignwrite**

Write a protassign input file with the current settings.

Syntax:

**protassignwrite**

**pspalignaddanchor**

Adds an anchor at the given position for Edit Alignment.

Syntax:

**pspalignaddanchor** < anchor position >

Operands:

< anchor position >

The position at which to add an anchor.

**pspaligndeleteanchor**

Deletes the anchor at the given position for Edit Alignment.

Syntax:

**pspaligndeleteanchor** < anchor position >

Operands:

< anchor position >

The position at which to delete an anchor.

## **pspaligninsertgaps**

Inserts gaps into the sequence at the given position.

Syntax:

**pspaligninsertgaps** *number=⟨ n ⟩ position=⟨ n ⟩ ⟨ sequence name ⟩*

Options:

*number* This is the number of gaps to insert.

Valid values: integers

Default value: **1**

Minimum: 1

*position* This is the position at which to insert a gap.

Valid values: integers

Default value: **1**

Minimum: 1

Operands:

⟨ sequence name ⟩

The name of the sequence to insert gaps into.

## **pspalignlockgaps**

Locks gaps for all alignments in the Edit Alignment step.

Syntax:

**pspalignlockgaps**

## **pspalignmoveleft**

Moves the alignment left freely starting at the given position. This will shift gaps from the left of the given position to the right of the given position.

Syntax:

**pspalignmoveleft** *number=⟨ n ⟩ position=⟨ n ⟩ ⟨ sequence name ⟩*

Options:

*number* This is the number of spaces to move left.

Valid values: integers

Default value: **1**

Minimum: 1

*position* This is the position at which to move left.

Valid values: integers

Default value: **1**

Minimum: 1

Operands:

⟨ sequence name ⟩

The name of the sequence to move left.

**pspalignmoveleftblock**

Moves the alignment left as a block starting at the given position. This will close up gaps to the left of the given position while preserving them to the right.

Syntax:

**pspalignmoveleftblock** *number=⟨ n ⟩ position=⟨ n ⟩ ⟨ sequence name ⟩*

Options:

*number* This is the number of spaces to move left.

Valid values: integers

Default value: **1**

Minimum: 1

*position* This is the position at which to move left.

Valid values: integers

Default value: **1**

Minimum: 1

Operands:

`< sequence name >`  
The name of the sequence to move left.

## **pspalignmoveright**

Moves the alignment right freely starting at the given position. This will shift gaps from the right of the given position to the left of the given position. If necessary, this will open up gaps to the left of the given position.

Syntax:

**pspalignmoveright** *number=*`< n >` *position=*`< n >` `< sequence name >`

Options:

*number*      This is the number of spaces to move right.

Valid values:      integers

Default value:      **1**

Minimum:      1

*position*      This is the position at which to move right.

Valid values:      integers

Default value:      **1**

Minimum:      1

Operands:

`< sequence name >`

The name of the sequence to move right.

## **pspalignstructures**

Runs 'structalign' to align the selected template structures in the Find Homologs step.

Syntax:

**pspalignstructures**

**pspalignunlockgaps**

Unlocks gaps for all alignments in the Edit Alignment step.

Syntax:

**pspalignunlockgaps**

**pspbstogglehetatom**

Toggles the given hetatom between included and excluded.

Syntax:

**pspbstogglehetatom** <hetatom name>

Operands:

<hetatom name>

The name of the hetatom to toggle.

**pspbuildbackbone**

Runs the build backbone backend.

Syntax:

**pspbuildbackbone**

**pspbuildstructure**

Runs the build structure backend.

Syntax:

### **pspbuildstructure**

### **pspexcludetable1row**

Excludes a composite structure from the table shown in the Build Backbone step. Excludes a structure, for the specified row in the input table shown in the current Structure Prediction step, from the Sequence Viewer (and Workspace). This applies to the Refine Backbone step.

Syntax:

**pspexcludetable1row** <row\_number>

Operands:

<row\_number>

The row number in the table which is to be excluded.

### **pspexcludetablerow**

Excludes a composite structure from the table shown in the Build Backbone step. Excludes a structure, for the specified row in the table shown in the current Structure Prediction step, from the Sequence Viewer (and Workspace). This applies to the Fold Recognition, Build Backbone, Refine Backbone, and Refine Structure steps (which have structure or template tables, and the ability to specify rows in Workspace independently from selected rows).

Syntax:

**pspexcludetablerow** <row\_number>

Operands:

<row\_number>

The row number in the table which is to be excluded.

## **pspexportalignment**

exports alignments selected into a file.

Syntax:

**pspexportalignment** <filename>

Operands:

<filename>

The name of the file to which the alignments are to be written.

## **pspfindfamily**

Finds the family for the query sequence currently active in the sequence viewer.

Syntax:

**pspfindfamily**

## **pspfindhomologs**

Runs Blast to find homologs for the query sequence currently active in the sequence viewer.

Syntax:

**pspfindhomologs**

## **pspfoldrecognitionoptions**

Sets some options associated with the Fold Recognition step.

Syntax:

### **pspfoldrecognitionoptions** *max\_results=⟨ n ⟩*

Options:

*max\_results*

Specifies how many results will be shown in the fold recognition results table.

Valid values: integers

Default value: **100**

### **pspfoldrecognitionsearch**

Runs the fold recognition search on the current query sequence.

Syntax:

### **pspfoldrecognitionsearch**

### **pspimportalignment**

imports alignments in a file and applies to the templates. since there can be more than one alignment stored in a file, the command gives messages on which templates have been altered.

Syntax:

### **pspimportalignment** ⟨ filename ⟩

Operands:

⟨ filename ⟩

The name of the file from which the alignments are to be read.

### **pspimporthomolog**

Imports the sequences from a file as homologs to the current query sequence.

Syntax:

**pspimporthomolog** <filename>

Operands:

<filename>

The name of the file from which the sequences are to be read.

**pspimportssp**

Adds the secondary structure prediction from a file to those for the current query sequence.

Syntax:

**pspimportssp** <filename>

Operands:

<filename>

The name of the file from which the SSP is to be read.

**pspincludetable1row**

Includes a structure from the input table shown in the current Structure Prediction step into the Sequence Viewer (and Workspace), independent of selection. This applies to the Refine Backbone step.

Syntax:

**pspincludetable1row** <row\_number>

Operands:

<row\_number>

The row number in the table which is to be included.

## **pspincludablerow**

Includes a structure from the table shown in the current Structure Prediction step into the Sequence Viewer (and Workspace). This applies to the Fold Recognition, Build Backbone, Refine Backbone, and Refine Structure steps (which have structure or template tables, and the ability to specify rows in Workspace independently from selected rows).

Syntax:

**pspincludablerow** <row\_number>

Operands:

<row\_number>

The row number in the table which is to be included.

## **pspminimizationresidues**

Defines a set of atoms for minimization refinement.

Syntax:

**pspminimizationresidues** <ASL>

Operands:

<ASL>

The ASL expression which defines the atoms that will be used to define the residues for minimization refinement.

## **pspoptimizealignment**

optimizes alignment of a template sequence.

Syntax:

**pspoptimizealignment** <rowindex>

Operands:

<rowindex>

The row index of the sequence which is aligned

## **psprefinebackbone**

Runs the refine backbone backend.

Syntax:

**psprefinebackbone**

## **psprefinestructure**

Runs the structure refinement program with the currently set refinement options on the currently selected structure.

Syntax:

**psprefinestructure**

## **psprsdefaulthelixloops**

Rebuilds the Helices & Loops table for structure refinement with the default loop and helix refinement options for the currently selected structure.

Syntax:

**psprsdefaulthelixloops**

## **psprshelixlooptoggrefine**

Toggle refinement of loop or helix.

Syntax:

**psprshelixlooptoggerefine** <row\_index>

Operands:

<row\_index>

The row index for the loop or helix in the Helices & Loops table that is to be toggled for refinement.

**psprshelixoptions**

Specifies the refinement options for the currently selected helix in the Helices & Loops table.

Syntax:

```
psprshelixoptions roll_range=<x> roll_resolution=<x>
    set_roll=yes | no set_shift=yes | no set_tilt=yes | no
    set_trans=yes | no shift_range=<x> shift_resolution=<x>
    tilt_range=<x> tilt_resolution=<x> trans_range=<x>
    trans_resolution=<x>
```

Options:

*roll\_range* Range, in degrees, for helix roll.

Valid values: reals

Default value: **20**

*roll\_resolution*

Resolution, in degrees, for helix roll.

Valid values: reals

Default value: **20**

*set\_roll* Set the helix roll range and resolution.

Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **false**

*set\_shift* Set the helix shift range and resolution.

Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **true**

*set\_tilt* Set the helix tilt range and resolution.

Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **false**

*set\_trans* Set the helix translation range and resolution.  
Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **false**

*shift\_range*  
Range, in angstroms, for helix shift.  
Valid values: reals  
Default value: **2**

*shift\_resolution*  
Resolution, in angstroms, for helix shift.  
Valid values: reals  
Default value: **2**

*tilt\_range* Range, in degrees, for helix tilt.  
Valid values: reals  
Default value: **5**

*tilt\_resolution*  
Resolution, in degrees, for helix tilt.  
Valid values: reals  
Default value: **5**

*trans\_range*  
Range, in angstroms, for helix translation.  
Valid values: reals  
Default value: **2**

*trans\_resolution*  
Resolution, in angstroms, for helix translation.  
Valid values: reals  
Default value: **2**

## psprsloopoptions

Specifies the refinement options for the currently selected loop in the Helices & Loops table.

Syntax:

**psprsloopoptions** *ca\_max=⟨x⟩ overlap\_min=⟨x⟩*  
*restrict\_ca=yes | no sphere\_size=⟨x⟩ use\_sphere=yes | no*

Options:

*ca\_max* Maximum CA atom movement, in angstroms, from initial.

Valid values: reals

Default value: **3**

*overlap\_min*

Minimum overlap, in angstroms, from initial.

Valid values: reals

Default value: **0.7**

Minimum: 0.1

Maximum: 1.0

*restrict\_ca* Restrict loop CA atom movement.

Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **false**

*sphere\_size*

Distance from loop for sidechain inclusion, in angstroms.

Valid values: reals

Default value: **7.5**

*use\_sphere* Include nearby sidechains in loop refinement.

Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **true**

## psprunalign

Runs secondary structure prediction (if necessary) and then aligns the query to the active templates in Edit Alignment.

Syntax:

**psprunalign**

## pspruncreate

Creates the run with the given name.

Syntax:

**pspruncreate** <run name>

Operands:

<run name>

The name of the new run to create.

## **psprundelete**

Deletes the current run from the project.

Syntax:

**psprundelete**

## **psprunopen**

Opens the run with the given name.

Syntax:

**psprunopen** <run name>

Operands:

<run name>

The name of the run to open.

## **psprunrename**

Changes the current run's name to the given name.

Syntax:

**psprunrename** <run name>

Operands:

<run name>

The name to change the current run's name to.

**psprunsaveas**

Saves a copy of the current run under the given name.

Syntax:

**psprunsaveas** <run name>

Operands:

<run name>

The name of the run to save as.

**pspsecstructprediction**

Runs the available secondary structure prediction programs on the current query sequence.

Syntax:

**pspsecstructprediction**

**pspselectextendtable1row**

Extend the selection from the selected table row to joining up with an existing selection. This applies to the Refine Backbone step

Syntax:

**pspselectextendtable1row** <row\_number>

Operands:

<row\_number>

The row number in the table from which the selection is to begin.

**pspselectextendtablerow**

Extend the selection from the selected table row to joining up with an existing selection. This applies to the Find Homologs, Fold Recognition, Build Backbone, Refine Backbone, Edit Alignment and Refine Structure steps.

Syntax:

**pspselectextendtablerow** <row\_number>

Operands:

<row\_number>

The row number in the table from which the selection is to begin.

**pspselecthelixlooprow**

Select structure for refinement.

Syntax:

**pspselecthelixlooprow** <row\_index>

Operands:

<row\_index>

The row index for the loop or helix in the Helices & Loops table that is to be selected for refinement option editing.

## **pspselectonlytable1row**

Select a row from the input structure or template table shown in the current Structure Prediction step, deselecting all other rows. This applies to the Refine Backbone step.

Syntax:

**pspselectonlytable1row** <row\_number>

Operands:

<row\_number>

The row number in the table which is to be selected.

## **pspselectonlytablerow**

Select a row from the structure or template table shown in the current Structure Prediction step, deselecting all other rows. This applies to the Find Homologs, Fold Recognition, Build Backbone, Refine Backbone, Edit Alignment and Refine Structure steps.

Syntax:

**pspselectonlytablerow** <row\_number>

Operands:

<row\_number>

The row number in the table which is to be selected.

## **pspselectrscontext**

Specify the context for structure refinement. This command may be executed instead of showing the psp or refinement panels. Be cautious about using this commands while either of these panels is shown, as setting the wrong context will interfere with the execution of the refinement commands issued from the panel.

Syntax:

### **pspselectrscontext pspstep|standalone**

Operands:

pspstep|standalone

The context in which refine structure commands are executed.

### **pspselectrsrefinement**

Selects the current psp structure refinement type.

Syntax:

#### **pspselectrsrefinement <refine\_type>**

Operands:

<refine\_type>

The type of structure refinement to be performed. Value should be sidechains , helixloops , or minimization .

### **pspselecttable1row**

Selects a row from the input structure or template table shown in the current Structure Prediction step. This applies to the Refine Backbone step.

Syntax:

#### **pspselecttable1row <row\_number>**

Operands:

<row\_number>

The row number in the table which is to be selected.

### **pspselecttablerow**

Selects a row from the structure or template table shown in the current Structure Prediction step. This applies to the Find Homologs, Fold Recognition,

Build Backbone, Refine Backbone, Edit Alignment and Refine Structure steps.

Syntax:

**pspselecttablerow** <row\_number>

Operands:

<row\_number>

The row number in the table which is to be selected.

## **pspsequenceaddfile**

Adds the sequences from the given file to the Select Sequence step in PSP.

Syntax:

**pspsequenceaddfile** <file>

Operands:

<file>

The file name of the sequence file to add.

## **pspsequenceaddworkspace**

Adds the sequences from the Workspace to the Select Sequence step in PSP.

Syntax:

**pspsequenceaddworkspace**

## **pspsequencecrop**

Crops the given sequence to the given residue index, towards the closer end of the sequence.

Syntax:

**pspsequencecrop** *res=*< n > < sequence name > < residue index >

Options:

*res* This is the position to crop to  
Valid values: integers  
Default value: 1  
Minimum: 1

Operands:

< sequence name > < residue index >

The name of the sequence to crop.

## **pspsequenceselect**

Selects the given sequence number as input for the next step in PSP.

Syntax:

**pspsequenceselect** < sequence number >

Operands:

< sequence number >

The index of the sequence to select.

## **pspsequenceviewerexport**

Exports all of the visible sequences in the sequence viewer to the given file.

Syntax:

**pspsequenceviewerexport** < file name >

Operands:

< file name >

The name of the file to export the sequences to.

**pspsethelixloopresidues**

Specify residues to define loop or helix feature for structure refinement.

Syntax:

**pspsethelixloopresidues** <row\_index> <R1> <R2> <R3> <R4>

Operands:

<row\_index> <R1> <R2> <R3> <R4>

The row index for the loop or helix in the Helices & Loops table, followed by the strings for the four residue column values, as they are to appear in the table. For a loop, only two end residues (<R1> <R2>) are required, so the <R3> and <R4> values can be missing or left blank. For a helix, <R1> and <R4> specify the outer residues for the loops at the two ends of the helix, while <R2> and <R3> specify the ends of the helix itself. The format for the residue strings is the residue number followed by the insertion code, if any. For example, 12C represent residue number 12, insertion code C.

**pspsidechainresidues**

Defines a set of atoms for the side chain refinement.

Syntax:

**pspsidechainresidues** <ASL>

Operands:

<ASL>

The ASL expression which defines the atoms that will be used to define the residues for side chain refinement.

**pspsortbbtable**

Resort the Build Backbone table based on the data in the specified column

Syntax:

**pspsortbbtable** <column\_name>

Operands:

<column\_name>

The name of the column to be sorted.

**pspsortfindhomologstable**

Resort the find homologs table based on the data in the specified column

Syntax:

**pspsortfindhomologstable** <column\_name>

Operands:

<column\_name>

The name of the column to be sorted.

**pspsortfoldtable**

Resort the fold recognition table based on the data in the specified column

Syntax:

**pspsortfoldtable** <column\_name>

Operands:

<column\_name>

The name of the column to be sorted.

**pspsortrbtable**

Resort the Refine Backbone output structure table based on the data in the specified column

Syntax:

**pspsortrbtable** <column\_name>

Operands:

<column\_name>

The name of the column to be sorted.

### **pspsortrbtable1**

Resort the Refine Backbone composite structure table based on the data in the specified column

Syntax:

**pspsortrbtable1** <column\_name>

Operands:

<column\_name>

The name of the column to be sorted.

### **pspsortrstable**

Resort the Refine Structure table based on the data in the specified column

Syntax:

**pspsortrstable** <column\_name>

Operands:

<column\_name>

The name of the column to be sorted.

### **pspsspdelete**

Deletes the given secondary structure prediction.

Syntax:

**pspsspdelete** < SSP name >

Operands:

< SSP name >

The name of the secondary structure prediction to delete.

## **pspsssexport**

Exports the given secondary structure to the named file.

Syntax:

**pspsssexport** *ssp*=< text > < file name >

Options:

*ssp* This option is the name of the secondary structure prediction to export.

Valid values: text strings

Default value:

Operands:

< file name >

The name of the file to export to.

## **pspssprevert**

Reverts a modified secondary structure prediction back to its original form.

Syntax:

**pspssprevert** < SSP name >

Operands:

< SSP name >

The name of the secondary structure prediction to revert.

## **pspsspset**

Sets the given range of positions in the secondary structure position to the given code, which must be H , E , or - .

Syntax:

**pspsspset** *code=⟨text⟩ from=⟨n⟩ to=⟨n⟩ ⟨SSP name⟩*

Options:

<i>code</i>	This is the code for setting the secondary structure position to. It must be one of H , E , or - .  Valid values: text strings Default value:
<i>from</i>	This is the starting position for setting codes.  Valid values: integers Default value: 1 Minimum: 1
<i>to</i>	This is the ending position for setting codes.  Valid values: integers Default value: 1 Minimum: 1

Operands:

⟨SSP name⟩

The name of the step to set data for.

## **pspstepforward**

Moves forward to the next named step. Deletes any steps after the current step and copies data forward to create the specified step.

Syntax:

**pspstepforward** ⟨step name⟩

Operands:

⟨step name⟩

The name of the step to switch to.

## **pspstepgoto**

Moves to an existing step in the current run.

Syntax:

**pspstepgoto** <step name>

Operands:

<step name>

The name of the step to switch to.

## **pspstructureaddentry**

Create project entry from selected psp structure. This operation is only permitted for the Refine Backbone and Refine Structure steps.

Syntax:

**pspstructureaddentry** <row\_index> <entry\_name>

Operands:

<row\_index> <entry\_name>

The optional operands are the row index of the psp refined structure and the entry base name from which a unique name will be generated for the new project entry. The row is the full, not the visible, row index into the table of structures for the current Structure Prediction step. If the operands are missing, the selected row(s) from the table will be used to create project entries, using the row structure name to generate the entry name.

## **psptemplatesetregion**

Sets the region for the given template.

Syntax:

**psptemplatesetregion** *end=⟨n⟩ start=⟨n⟩ ⟨ template name ⟩*

Options:

*end* This is the position at which the region ends.

Valid values: integers

Default value: **1**

Minimum: 1

*start* This is the position at which to start the region.

Valid values: integers

Default value: **1**

Minimum: 1

Operands:

⟨ template name ⟩

The name of the template to set a region for.

**psptmplsecstructprediction**

Runs the available secondary structure prediction programs on the template sequence identified by row number

Syntax:

**psptmplsecstructprediction** ⟨ row\_number ⟩

Operands:

⟨ row\_number ⟩

The row number in the table which is to be selected.

**pspunselectable1row**

Unselects a row from the input table shown in the current Structure Prediction step. This applies to the Refine Backbone step.

Syntax:

**pspunselecttable1row** <row\_number>

Operands:

<row\_number>

The row number in the table which is to be unselected.

**pspunselecttablerow**

Unselects a row from the table shown in the current Structure Prediction step. This applies to the Find Homologs, Fold Recognition, Build Backbone, Refine Backbone, and Edit Alignment steps (which support multiple row selection).

Syntax:

**pspunselecttablerow** <row\_number>

Operands:

<row\_number>

The row number in the table which is to be unselected.

**pspupdatescores**

Updates Scores of all the alignments

Syntax:

**pspupdatescores**

**pyeval**

This is a standard alias for **pythoneval** (see [pythoneval], page 403).

## **pyimp**

This is a standard alias for **pythonimport** (see [pythonimport], page 403).

## **pyrun**

This is a standard alias for **pythonrun** (see [pythonrun], page 404).

## **pythoneval**

Evaluate the python expression using the built-in Python interpreter.  
KEY\_OPERAND\_SYNOPSIS: <python expression>

Syntax:

### **pythoneval**

Aliases:

**pyeval** (see [pyeval], page 402)

## **pythonimport**

Imports the specified python module. If the module has already been imported then it will be reloaded. Under normal usage this command is not needed as pythonrun automatically imports the module specified as its operand. However during development of modules it is useful to be able to reload them using pythonimport  
KEY\_OPERAND\_SYNOPSIS: <module>

Syntax:

### **pythonimport**

Aliases:

**pyimp** (see [pyimp], page 403)

## **pythonrun**

Runs the function in <function name> in module <modulename>  
KEY\_OPERAND\_SYNOPSIS: <function name> [function parameters]

Syntax:

### **pythonrun**

Aliases:

**pyrun** (see [pyrun], page 403)

## **qikprop**

This keyword is used to set various options associated with running QikProp jobs.

Syntax:

**qikprop** *fastmode=yes | no* *structure\_source=selected\_entries | workspace | file*

Options:

*fastmode* The mode in which QP job runs. (fast/slow)

Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **false**

*structure\_source*

Whether to use the selected entries in the current project or what is in the workspace as input for the job.

Valid values: selected\_entries  
workspace  
file

Default value: **selected\_entries**

## **qsarmarkerdump**

Print out the current option values of the QSAR marker command.

Syntax:

**qsarmarkerdump**

## **qsarmarkersettings**

Set graphical data of Phase QSAR markers.

Syntax:

```
qsarmarkersettings ambient=<x> diffuse=<x> effects=allclass |  
    selectedclass emission=<x> negativecoefficient=<x>  
    numberpls=<n> positivecoefficient=<x> roundingeffect=<x>  
    selectedatomclass=<text> shininess=<x> specular=<x>  
    step=<n> transparency=<x> volumeoccupied=workspacelig |  
    qsarmodel
```

Options:

*ambient* Set material property - ambient, to its red, green, and blue components, for front face.

Valid values:    reals  
Default value: **0.4**  
Minimum:        0.0  
Maximum:        1.0

*diffuse* Set material property - diffuse, to its red, green, and blue components, for front face.

Valid values:    reals  
Default value: **0.4**  
Minimum:        0.0  
Maximum:        1.0

*effects* Set the QSAR visualization option of viewing effects.

Valid values:    allclass  
                  selectedclass  
Default value: **allclass**

*emission* Set material property - emission, to its red, green, and blue components, for front face.

Valid values:    reals  
Default value: **0.1**  
Minimum:        0.0  
Maximum:        1.0

*negativecoefficient*

Set the QSAR visualization option of negative coefficient threshold.

Valid values:   reals

Default value:   **-0.044**

Maximum:       0.0

*numberpls* Set the QSAR visualization option of number of PLS factors.

Valid values:   integers

Default value:   **1**

Minimum:       1

*positivecoefficient*

Set the QSAR visualization option of positiv coefficient threshold.

Valid values:   reals

Default value:   **0.044**

Minimum:       0.0

*roundingeffect*

This determines the rounding effect of edges of a cell, for front face.

Valid values:   reals

Default value:   **5**

Minimum:       0.0

*selectedatomclass*

Set the QSAR visualization option of selected atom class.

Valid values:   text strings

Default value:   **D**

*shininess* Set material property - shininess, for front face.

Valid values:   reals

Default value:   **80**

Minimum:       0.0

Maximum:       128.0

*specular* Set material property - specular, to its red, green, and blue components, for front face.

Valid values:   reals

Default value:   **0.1**

Minimum:       0.0

Maximum:       1.0

*step* The step domain tolerance of cells.

Valid values:   integers

Default value:   **3**

Minimum:       1

*transparency*

The transparency of QSAR markers.

Valid values:   reals

Default value:   **50**

Minimum:       0.0

Maximum:       100.0

*volumeoccupied*

Set the QSAR visualization option of viewing volume occupied.

Valid values:   workspacelig

                  qsarmodel

Default value:   **workspacelig**

## qsiteion

Defines an ion as part of the QM region for an Impact QSite simulation

Syntax:

**qsiteion** *basis*=⟨ text ⟩ ⟨ atom\_num ⟩

Options:

*basis*       The basis set for the ligand to be included in the QM region of an Impact QSite simulation.

Valid values:   text strings

Default value:   **lacvp\***

Operands:

⟨ atom\_num ⟩

The atom number of an ion to be included in the QM region in a Impact QSite simulation.

## qsiteresidue

Defines a residue as part of the QM region for an Impact QSite simulation

Syntax:

**qsiteresidue** *basis*=⟨text⟩ *cuttype*=⟨n⟩  
⟨chain⟩:⟨molnum⟩:⟨resnum⟩:⟨inscode⟩

Options:

*basis* The basis set for the residue to be included in the QM region of an Impact QSite simulation.

Valid values: text strings  
Default value: **lacvp\***

*cuttype* The cut type for the residue to be included in the QM region of an Impact QSite simulation.

Valid values: integers  
Default value: **1**  
Minimum: **0**  
Maximum: **5**

Operands:

⟨chain⟩:⟨molnum⟩:⟨resnum⟩:⟨inscode⟩

The number of a residue to be included in the QM region in a Impact QSite simulation.

## qsiteset

Settings associated with QSite simulations in Impact.

Syntax:

**qsiteset** *charge*=⟨n⟩ *maxiter*=⟨n⟩ *method*=dft | hf | lmp2  
*multiplicity*=⟨n⟩ *numberprocessors*=⟨n⟩  
*optimization*=singlepoint | minimization | transitionstate  
*optimize*=yes | no *options*=⟨text⟩ *pathfraction*=⟨x⟩  
*product\_entry*=⟨text⟩ *reactant\_entry*=⟨text⟩  
*tsguess\_entry*=⟨text⟩ *tsmethod*=standard | lst | qst

Options:

*charge* The total charge on the QM part in an Impact QSite simulation.  
Valid values: integers  
Default value: **0**

*maxiter* The maximum number of iterations for the QM optimization.  
Valid values: integers  
Default value: **100**

*method*      The QM Method used in a QSite job in Maestro

Valid values:    dft  
                  hf  
                  lmp2

Default value: **dft**

*multiplicity*

The multiplicity of the QM part in an Impact QSite simulation.

Valid values:    integers  
Default value: **1**

*numberprocessors*

The number of processors to be used for the Jaguar Part of the calculation.

Valid values:    integers  
Default value: **1**

*optimization*

What type of calculation is to be performed in the QM part.

Valid values:    singlepoint  
                  minimization  
                  transitionstate

Default value: **minimization**

*optimize*

[NOTE: This option is no longer used.] An option which determines if the QM part will be geometry optimized during a QSite job.

Valid values:    boolean (true|false; yes|no; y|n; on|off)  
Default value: **true**

*options*

Any additional Jaguar options which can be used during the Impact QSite job.

Valid values:    text strings  
Default value: **iacc=1 vshift=1.0 maxit=100**

*pathfraction*

The fraction of the path between the reactant and product which the TS is along.

Valid values:    reals  
Default value: **0.5**  
Minimum:       -0.000000  
Maximum:       1.0000001

*product\_entry*

The name of the entry which represents the product in a transition state calculation.

Valid values:    text strings  
Default value:

*reactant\_entry*

The name of the entry which represents the entry in a transition state calculation.

Valid values: text strings

Default value:

*tsguess\_entry*

The name of the entry which represents the transition state guess in a transition state calculation.

Valid values: text strings

Default value:

*tsmethod* For a transition state calculation, how that is to be performed.

Valid values: standard

lst

qst

Default value: **standard**

## quit

Quit the program. To quit issue just the quit command without any options.

Syntax:

**quit** *confirm=yes | no*

Options:

*confirm* If this option has been set to “false” then the program will exit without prompting the user in any way. So, for example, the user is not prompted to save any changed macros nor will the Quit panel be displayed. Note that invoking quit with any options only sets the option it does not also try to quit Maestro.

Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **true**

## read

This is a standard alias for **fileread** (see [fileread], page 138).

## **readposefile**

Read the receptor and ligand poses from pose file in the pose viewer.

Syntax:

**readposefile** <file\_name>

Operands:

<file\_name>

The name of the file from which the structures will be read. If no name is specified, the current pose list will be cleared.

## **readpotential**

Read potential settings from a command file.

Syntax:

**readpotential** <file\_name>

Operands:

<file\_name>

The name of the file from which the potential settings will be read. If no name is specified, the default settings will be used.

## **reagentprep**

Options for Reagent Preparation jobs.

Syntax:

```
reagentprep gen_conform=⟨n⟩ gen_ionization=yes | no  
    gen_stereo=⟨n⟩ gen_tautomers=yes | no  
    group_name_long=⟨text⟩ group_name_short=⟨text⟩  
    input_file=⟨text⟩ ph=⟨x⟩ ph_tolerance=⟨x⟩  
    sd_title_property=⟨text⟩ sd_title_source=molecule_name |  
    property structure_source=selected_entries | workspace | file
```

Options:

*gen\_conform*

The percentage of generating low energy ring . conformations.

Valid values: integers

Default value: **1**

Minimum: 1

*gen\_ionization*

An option which allows generating ionization.

Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **true**

*gen\_stereo* The percentage of generating stereoisomers.

Valid values: integers

Default value: **10**

Minimum: 1

*gen\_tautomers*

An option which allows generating tautomers.

Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **true**

*group\_name\_long*

The long name of selected functional group for reagent preparation job.

Valid values: text strings

Default value:

*group\_name\_short*

The short name of selected functional group for reagent preparation job.

Valid values: text strings

Default value:

*input\_file* The name of the structure input file.

Valid values: text strings

Default value:

*ph* The reagent ionization pH value.

Valid values:    real  
Default value: **7**  
Minimum:        0.0  
Maximum:        14.0

*ph\_tolerance*

The reagent ionization pH tolerance.

Valid values:    real  
Default value: **2**  
Minimum:        0.0  
Maximum:        7.0

*sd\_title\_property*

The property to be used to construct titles for reagents from a SD format file, if *sd\_title\_source* is 2 - property.

Valid values:    text strings  
Default value:

*sd\_title\_source*

The source of titles for reagents from a SD format file (1 - molecule\_name or 2 - property).

Valid values:    molecule\_name  
                  property  
Default value: **molecule\_name**

*structure\_source*

Whether to use the selected entries in the current project, or what is in the workspace, or a specified file with multiple structures as structure input for the job.

Valid values:    selected\_entries  
                  workspace  
                  file  
Default value: **file**

## **refinestart**

Start a Refine input file with the current settings.

Syntax:

## **refinestart**

### **refinewrite**

Write a Refine input file with the current settings.

Syntax:

### **refinewrite**

## **rename**

Rename a named object. The object type is the same as the command which is used to create that type of object.

For example to rename a set named “alpha” use: `rename set alpha beta .`

Syntax:

**rename** <object\_type> <current\_object\_name> <new\_object\_name>

Operands:

<object\_type> <current\_object\_name> <new\_object\_name>

The first operand is the name of the existing object.

## **renameproperty**

This is a standard alias for **propertyrename** (see [propertyrename], page 371).

## **repall**

Set global representation properties

Syntax:

**repall** *ballhresolution=⟨n⟩ balllresolution=⟨n⟩ ballsize=⟨x⟩ border=auto | on | true | yes | off | false | no borderscale=⟨x⟩ bstyle=split | blend cpkhresolution=⟨n⟩ cpklresolution=⟨n⟩ cpksize=⟨x⟩ resolution=high | low rstyle=multiple | thick simplifymoving=yes | no smooth=yes | no stickradius=⟨x⟩ tuberadius=⟨x⟩ wirethickness=⟨n⟩*

Options:

*ballhresolution*

Set Ball high resolution  
 Valid values: integers  
 Default value: **16**  
 Minimum: 4  
 Maximum: 53

*balllresolution*

Set Ball resolution  
 Valid values: integers  
 Default value: **4**  
 Minimum: 4  
 Maximum: 53

*ballsize* Percentage to draw balls out at

Valid values: reals  
 Default value: **25**  
 Minimum: 4.0  
 Maximum: 200.0

*border* Set drawing of bond order to be on, off, or automatically determined by viewing scale.

Valid values: auto  
 on  
 true  
 yes  
 off  
 false  
 no  
 Default value: **auto**

*borderscale*

The drawing scale at which bond orders will be displayed

Valid values: reals  
 Default value: **30**

*bstyle* Set the bond style

Valid values: split  
blend  
Default value: **split**

*cpkhresolution*

Set CPK high resolution  
Valid values: integers  
Default value: **20**  
Minimum: 8  
Maximum: 57

*cpklresolution*

Set CPK low resolution  
Valid values: integers  
Default value: **8**  
Minimum: 8  
Maximum: 57

*cpksize* Percentage to draw CPK spheres out at

Valid values: reals  
Default value: **85**  
Minimum: 4.0  
Maximum: 200.0

*resolution* Set the overall resolution for drawing molecules. low or high.

Valid values: high  
low  
Default value: **high**

*rstyle* Set the bond render style

Valid values: multiple  
thick  
Default value: **multiple**

*simplifymoving*

Enable/disable use of simplified moving representation  
Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **true**

*smooth* Enable/disable line and polygon antialiasing

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **true**

*stickradius*

Radius of sticks in Angstroms  
Valid values: reals  
Default value: **0.15**

Minimum: 0.01  
Maximum: 1.0

*tuberadius* Radius of tubes in Angstroms

Valid values: reals  
Default value: **0.25**  
Minimum: 0.01  
Maximum: 1.0

*wirethickness*

Thickness of wireframe lines

Valid values: integers  
Default value: **2**  
Minimum: 1  
Maximum: 20

## repatom

Change the representation used to display a group of atoms in the main structure window.

Syntax:

**repatom** *rep=none | circle | cpk | ballnstick <ASL>*

Options:

*rep* Type of atom representation  
Valid values: none  
circle  
cpk  
ballnstick  
Default value: **none**

Operands:

*<ASL>*

A string in the atom specification language. All atoms which match this specification will have their representation changed.

## **repatombonds**

Set representation of all of atoms' bonds. Uses bond rep specified in the repbond command.

Syntax:

**repatombonds** ⟨ ASL ⟩

Operands:

⟨ ASL ⟩

A string in the atom specification language. All atoms which match this string will have the representation of all bonds to them changed.

## **repbond**

Change the representation used to draw an on-screen bond.

Syntax:

**repbond** *rep=none | wire | tube* ⟨ atom1 ⟩ ⟨ atom2 ⟩

Options:

<i>rep</i>	Type of bond representation
Valid values:	none wire tube
Default value:	<b>wire</b>

Operands:

⟨ atom1 ⟩ ⟨ atom2 ⟩

The operands represent the numbers of the two atoms which define the bond to have its representation changed.

## **repdefault**

Set representation of all atoms and bonds

Syntax:

**repdefault** *style=wire | cpk | ballnstick | tube*

Options:

*style*      Default representation  
Valid values:    wire  
                  cpk  
                  ballnstick  
                  tube  
Default value: **wire**

## replacefromhold

This command replaces the on-screen structure with a copy of the the structure from the specified hold set.

Syntax:

**replacefromhold** *<hold\_name>*

Operands:

*<hold\_name>*

The name of the hold. This must be the name which was specified when the hold was created using the “hold” command.

## repquick

Set representation of all atoms and bonds

Syntax:

**repquick** *default=default | wire | cpk | ballnstick | tube*  
*style=default | wire | cpk | ballnstick | tube [update]*

Options:

*default*      Default representation

Valid values:	default wire cpk ballnstick tube
Default value:	<b>wire</b>
<i>style</i>	Default representation
Valid values:	default wire cpk ballnstick tube
Default value:	<b>default</b>

Operands:

[update]

When update is present the currently set representation will be applied to the on-screen structure.

## **resetcsearch**

Deletes all the variables used in a conformational search.

Syntax:

## **resetcsearch**

## **residuename**

Set the residue name for all atoms which match the ASL specification.

Syntax:

## **residuename** <PDBNAME> <ASL>

Operands:

<PDBNAME> <ASL>

The first operand is the PDB residue name which will be used for all atoms which match the specification. The second operand is a valid ASL string which defines the set of atoms which are to have their residue names changed.

## **residuenumber**

Set the residue number for all atoms which match the ASL specification.

Syntax:

**residuenumber** <res\_num> <ASL>

Operands:

<res\_num> <ASL>

The first operand is an integer, optionally followed by a single-character insertion code, that specifies the residue number for all the atoms which match the ASL specification. If the trailing alphabetic character is omitted, the residue insertion code will be set blank. The second operand must be a valid ASL string which specifies all the atoms to have their residue number changed.

## **residuerenumber**

Renumber the residues, starting with the starting number, for all residues which match the ASL specification.

Syntax:

**residuerenumber** <starting\_res\_num> <ASL>

Operands:

<starting\_res\_num> <ASL>

The first operand is an integer which represents the starting residue number. The second operand must be a valid ASL string which specifies a set of residues to renumber.

## **restorepanels**

Restores panel locations

Syntax:

## **restorepanels**

## **retype**

Change the atom type of the atom number specified by the operand to whatever type or element has previously been made current with the atom command.

Syntax:

## **retype <atom\_num>**

Operands:

**<atom\_num>**

An atom number representing the atom which is to have its type changed to the current type.

## **ribbon**

Creates a new Ribbon.

Syntax:

**ribbon** *ambient*=⟨x⟩ *ambientback*=⟨x⟩ *calphalinewidth*=⟨n⟩  
*calphatubesteps*=⟨n⟩ *calphatubewidth*=⟨x⟩ *color*=black | gray  
| darkblue | blue | lightblue | aquamarine | turquoise |  
springgreen | darkgreen | green | limegreen | yellowgreen |  
yellow | orange | maroon | red | pink | plum | purple |  
bluepurple | white *curvedlinesteps*=⟨n⟩ *curvedlinewidth*=⟨n⟩  
*diffuse*=⟨x⟩ *diffuseback*=⟨x⟩ *display*=ribbonsonly | atomsonly  
| both *emission*=⟨x⟩ *emissionbackblue*=⟨x⟩  
*emissionbackgreen*=⟨x⟩ *emissionbackred*=⟨x⟩  
*helixcolor*=onecolor | twocolors *resolution*=high | low  
*ribbonendweight*=⟨x⟩ *ribbonhasthick*=yes | no  
*ribbonsteps*=⟨n⟩ *ribbonthick*=⟨x⟩ *ribbonweight*=⟨x⟩  
*ribbonwidth*=⟨x⟩ *scheme*=constant | secondarystructure |  
chain | calphaatom | residuecharge | residueproperty |  
residuestype | residueposition | entry *shininess*=⟨x⟩  
*shininessback*=⟨x⟩ *specular*=⟨x⟩ *specularback*=⟨x⟩  
*sphereslices*=⟨n⟩ *spherestacks*=⟨n⟩ *strandarrowweight*=⟨x⟩  
*strandarrowwidth*=⟨x⟩ *strandendweight*=⟨x⟩  
*strandendweight1*=⟨x⟩ *strandsteps*=⟨n⟩ *strandthick*=⟨x⟩  
*strandwidth*=⟨x⟩ *strandwidth*=⟨x⟩ *style*=none | cartoon |  
ribbon | tube | thintube | curvedline | calphaline | calphatube  
*thintubesteps*=⟨n⟩ *thintubeweight*=⟨x⟩ *thintubewidth*=⟨x⟩  
*tubesteps*=⟨n⟩ *tubeweight*=⟨x⟩ *tubewidth*=⟨x⟩  
⟨ ASL-definition ⟩

Options:

*ambient* Set material property - ambient, to its red, green, and blue components, for front face.  
Valid values: reals  
Default value: **0.5**  
Minimum: 0.0  
Maximum: 1.0

*ambientback* Set material property - ambient, to its red, green, and blue components, for back face.  
Valid values: reals  
Default value: **0.2**  
Minimum: 0.0  
Maximum: 1.0

*calphalinewidth* Set linewidth for drawing CA Trace ribbons.  
Valid values: integers  
Default value: **2**

Minimum: 1  
Maximum: 40

*calphatubesteps*

Set the steps of drawing ribbon CA Tube Trace from one node point (mapped from one CA atom) to another. In the both U and V directions. It is used when the resolution option is HIGH.

Valid values: integers  
Default value: **3**  
Minimum: 1  
Maximum: 10

*calphatubewidth*

Set CA tube width for drawing CA Trace Tube ribbons.

Valid values: reals  
Default value: **0.4**  
Minimum: 0.001

*color*

An option which controls the ribbon constant color.

Valid values: black  
gray  
darkblue  
blue  
lightblue  
aquamarine  
turquoise  
springgreen  
darkgreen  
green  
limegreen  
yellowgreen  
yellow  
orange  
maroon  
red  
pink  
plum  
purple  
bluepurple  
white  
Default value: **green**

*curvedlinesteps*

Set the steps of drawing ribbon Curved Line from one node point (mapped from one CA atom) to another. It is used when the resolution option is HIGH.

Valid values: integers  
 Default value: **9**  
 Minimum: 1  
 Maximum: 100

*curvedlinewidth*

Set curve width for drawing Curved Line ribbons.

Valid values: integers  
 Default value: **2**  
 Minimum: 1  
 Maximum: 40

*diffuse*

Set material property - diffuse, to its red, green, and blue components, for front face.

Valid values: reals  
 Default value: **0.4**  
 Minimum: 0.0  
 Maximum: 1.0

*diffuseback*

Set material property - diffuse, to its red, green, and blue components, for back face.

Valid values: reals  
 Default value: **0.2**  
 Minimum: 0.0  
 Maximum: 1.0

*display*

An option which controls whether the atoms which define the ribbon are to be shown. The three options are: Ribbon Only, Atoms Only, and Both.

Valid values: ribbononly  
 atomsonly  
 both  
 Default value: **ribbononly**

*emission*

Set material property - emission, to its red, green, and blue components, for front face.

Valid values: reals  
 Default value: **0.05**  
 Minimum: 0.0  
 Maximum: 1.0

*emissionbackblue*

Set material property - emission, to its blue component, for back face. Emission R, G, B values are used to control the color of back face. The greater the values, the lighter the back face.

Valid values: reals

Default value: **0.55**  
Minimum: 0.0  
Maximum: 1.0

*emissionbackgreen*

Set material property - emission, to its green component, for back face. Emission R, G, B values are used to control the color of back face. The greater the values, the lighter the back face.

Valid values: reals  
Default value: **0.52**  
Minimum: 0.0  
Maximum: 1.0

*emissionbackred*

Set material property - emission, to its red component, for back face. Emission R, G, B values are used to control the color of back face. The greater the values, the lighter the back face.

Valid values: reals  
Default value: **0.5**  
Minimum: 0.0  
Maximum: 1.0

*helixcolor* An option which controls whether ribbons in the helical part are to be colored with a single color or with two colors and the inside of the helix has a contrasting color.

Valid values: onecolor  
twocolors  
Default value: **twocolors**

*resolution* Set the resolution for drawing ribbons.

Valid values: high  
low  
Default value: **high**

*ribbonendweight*

This weight value is used to control the radius of the Ribbon ends. The greater the value, the smaller the radius.

Valid values: reals  
Default value: **0.6**  
Minimum: 0.001

*ribbonhasthick*

This bool value is used to determine which kind of ribbon will be drawn, a flat sheet or a solid ribbon has thick.

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **true**

*ribbonsteps*

Set the steps of drawing Ribbon ribbon from one node point (mapped from one CA atom) to another. In the both U and V directions. It is used when the resolution option is HIGH.

Valid values: integers

Default value: **3**

Minimum: 1

Maximum: 10

*ribbonthick*

This value is used to control the thick of ribbon, if the ribbon is display in solid ribbon.

Valid values: reals

Default value: **0.1**

Minimum: 0.02

Maximum: 2.0

*ribbonweight*

This weight value is used to control the radius of the rectangle cross-section of a strand ribbon. The great the value, the small the radius.

Valid values: reals

Default value: **3**

Minimum: 0.001

*ribbonwidth*

Set ribbon width for drawing Ribbon ribbons.

Valid values: reals

Default value: **1.3**

Minimum: 0.05

Maximum: 5.0

*scheme* An option which controls the ribbon coloring scheme.

Valid values: constant  
secondarystructure  
chain  
calphaatom  
residuecharge  
residueproperty  
residuetype  
residueposition  
entry

Default value: **secondarystructure**

*shininess* Set material property - shininess, for front face.

Valid values: reals

Default value: **80**  
Minimum: 0.0  
Maximum: 128.0

*shininessback*

Set material property - shininess, for back face.

Valid values: reals  
Default value: **20**  
Minimum: 0.0  
Maximum: 128.0

*specular* Set material property - specular, to its red, green, and blue components, for front face.

Valid values: reals  
Default value: **0.7**  
Minimum: 0.0  
Maximum: 1.0

*specularback*

Set material property - specular, to its red, green, and blue components, for back face.

Valid values: reals  
Default value: **0.2**  
Minimum: 0.0  
Maximum: 1.0

*sphereslices*

Set the slices of drawing sphere in the tube style. We use gluSphere() to draw a sphere. The higher the slices, the fine the sphere.

Valid values: integers  
Default value: **18**  
Minimum: 2

*spherestacks*

Set the stacks of drawing sphere in the tube style. We use gluSphere() to draw a sphere. The higher the stacks, the fine the sphere.

Valid values: integers  
Default value: **18**  
Minimum: 2

*strandarrowweight*

This weight value is used to control the radius of the strand arrow. The great the value, the small the radius.

Valid values: reals  
Default value: **10**

Minimum: 0.001

*strandarrowwidth*

This ratio value is used to control the width of the strand arrow.

Valid values: reals

Default value: **2**

Minimum: 1.0

*strandendweight*

This weight value is used to control the radius of the strand ends. The great the value, the small the radius.

Valid values: reals

Default value: **0.6**

Minimum: 0.001

*strandendweight1*

This weight value is used to control the radius of the strand ends in Cartoon. The great the value, the small the radius.

Valid values: reals

Default value: **50**

Minimum: 0.001

*strandsteps*

Set the steps of drawing ribbon Strand from one node point (mapped from one CA atom) to another. In the both U and V directions. It is used when the resolution option is HIGH.

Valid values: integers

Default value: **3**

Minimum: 1

Maximum: 10

*strandthick*

Set strand thick for drawing Strand ribbons.

Valid values: reals

Default value: **0.2**

Minimum: 0.02

Maximum: 2.0

*strandweight*

Ribbons are defined with NURBS curve or surfaces that are defined by 4D homogeneous coordinate (x, y, z, w) control points array or mesh. The weight w can push/pull away/towards the curve/surface part near the control point, by decreasing/increasing weight value. This weight value is used to control the radius of the rectangle cross-section of a strand ribbon. The great the value, the small the radius.

Valid values: reals

Default value: **5**  
Minimum: 0.001

*strandwidth*

Set strandwidth for drawing Strand ribbons.

Valid values: reals  
Default value: **1.5**  
Minimum: 0.05  
Maximum: 5.0

*style* An option which controls the ribbon style representation.

Valid values: none  
cartoon  
ribbon  
tube  
thintube  
curvedline  
calphaline  
calphatube  
Default value: **cartoon**

*thintubesteps*

Set the steps of drawing ribbon Thin Tube from one node point (mapped from one CA atom) to another. In the both U and V directions. It is used when the resolution option is HIGH.

Valid values: integers  
Default value: **3**  
Minimum: 1  
Maximum: 10

*thintubeweight*

This weight value is used to control the radius of the Thin Tube ends. The greater the value, the smaller the radius.

Valid values: reals  
Default value: **1**  
Minimum: 0.001

*thintubewidth*

Set thin tube width for drawing Thin Tube ribbons.

Valid values: reals  
Default value: **0.4**  
Minimum: 0.05  
Maximum: 5.0

*tubesteps* Set the steps of drawing ribbon Tube from one node point (mapped from one CA atom) to another. In the both U and V directions. It is used when the resolution option is HIGH.

Valid values: integers  
Default value: **3**  
Minimum: 1  
Maximum: 10

*tubeweight* This weight value is used to control the radius of the Tube ends.  
The greater the value, the smaller the radius.

Valid values: reals  
Default value: **1**  
Minimum: 0.001

*tubewidth* Set tube width for drawing Tube ribbons.

Valid values: reals  
Default value: **0.9**  
Minimum: 0.05  
Maximum: 5.0

Operands:

*<ASL-definition>*

The operand must be a valid string in the atom specification language. It will define which atoms are to have a ribbon drawn.

## **ribbondump**

Print out the current option values of the ribbon command.

Syntax:

## **ribbondump**

## **ringclosure**

A command which defines a ring closure to be used during a conformational search.

Syntax:

**ringclosure** *maximum*=⟨x⟩ *minimum*=⟨x⟩ ⟨atom1⟩ ⟨atom2⟩  
⟨atom3⟩ ⟨atom4⟩

Options:

*maximum* The maximum distance between the ends of the ring which will be accepted as a candidate for “closure”.

Valid values: reals

Default value: **2.5**

Minimum: 0.0

*minimum* The minimum distance between the ends of the ring which will be accepted as a candidate for “closure”.

Valid values: reals

Default value: **0.5**

Minimum: 0.0

Operands:

⟨atom1⟩ ⟨atom2⟩ ⟨atom3⟩ ⟨atom4⟩

The four atom numbers which define a point in a ring which is to be opened while new structures are generated in a conformational search. The actually opening takes place between the second and third atoms specified. Note that specifying a-b-c-d is the same as specifying d-c-b-a.

## **rotate**

Rotate in degrees whatever is specified in the transform set. This is either global (all atoms) or a local grouping defined via an ASL in the transform command.

Syntax:

**rotate** *x*=⟨x⟩ *y*=⟨x⟩ *z*=⟨x⟩ [reset]

Options:

*x* Amount in degrees to rotate in X

Valid values: reals

Default value: **0**

*y* Amount in degrees to rotate in Y

Valid values: reals

Default value: **0**

*z* Amount in degrees to rotate in Z

Valid values: reals

Default value: **0**

Operands:

[reset]

If reset is present the global rotation matrix will be reset.

## run

This is a standard alias for **scriptrun** (see [scriptrun], page 435).

## saveimage

Capture the current main structure window and save to an image file.

Syntax:

```
saveimage format=tiff | jpeg jpeg_quality=<n>
    png_compression=<n> png_gamma=<x> smooth=yes | no
    <file_name>
```

Options:

*format* Specifies the format of the saved image.

Valid values: tiff

jpeg

Default value: **tiff**

*jpeg-quality*

Quality of the JPEG file (1-100)

Valid values: integers

Default value: **75**

*png\_compression*

Compression ratio of png format

Valid values: integers

Default value: **6**

*png\_gamma*

Gamma of the image in png format

Valid values:   reals  
Default value:   **2.2**

*smooth*   Image saved will have smooth curves in it if enabled.  
Valid values:   boolean (true|false; yes|no; y|n; on|off)  
Default value:   **false**

Operands:

**⟨file\_name⟩**

The file where the image will be saved.

## **saveimageheight**

Sets the height of the image saved due to saveimage command  
KEY\_OPERADN\_SYNOPSIS: <height>

Syntax:

### **saveimageheight**

## **saveimagewidth**

Sets the width of the image saved due to saveimage command  
KEY\_OPERADN\_SYNOPSIS: <width>

Syntax:

### **saveimagewidth**

## **savelayout**

Save the size and position of all currently visible panels.

Syntax:

## **savelayout**

### **scriptlogfile**

Commands are by default logged to a temporary file which is deleted when the program terminates. If the “logfile” command is used to name the logfile then the commands will be logged to that file and that file will not be deleted when the program ends.

Syntax:

**scriptlogfile** *logging=yes | no <logfile\_name>*

Options:

*logging*      If this option is set to “false” then no logging will be done to a file or in memory.

Valid values:    boolean (true|false; yes|no; y|n; on|off)

Default value:   **true**

Operands:

*<logfile\_name>*

The name of the file to which the commands will be logged. The full name of the file, including any suffix must be included.

Aliases:

**logfile** (see [logfile], page 236)

## **scriptrun**

Run the script file whose name is given as the operands.

Syntax:

**scriptrun** *<script\_name>*

Operands:

*<script\_name>*

The name of the script which is to be executed. The full name, including any suffix must be given. If the file is not in the local directory then a full pathname must be given.

Aliases:

**run** (see [run], page 433)

## searchdbcongen

Defines settings for Find Matches to Hypothesis Generate Conformers job.

Syntax:

```
searchdbcongen field=mmffs | opls2005 incorporate=append | replace | ignore maxdist=<x> method=default | mixed numsteps=<n> postmaxiter=<n> postprocessing=yes | no premaxiter=<n> preprocessing=yes | no sampling=standard | rapid | complete | thorough solvation=gbsa | distance-dependent window=<x>
```

Options:

*field* This determines which force field mmffs|mmff|opls2001 is used. Currently we always use mmffs, so it will have only one option value.

Valid values: mmffs  
opls2005

Default value: **opls2005**

*incorporate*

This option controls the incorporation of the results (replace or append).

Valid values: append  
replace  
ignore

Default value: **replace**

*maxdist* Maximum distance between atoms in equal structures.

Valid values: reals

Default value: **2**

Minimum: 0.0

*method* This determines whether MacroModel uses the ligand torsion search method (default) or the mixed MCMM/LMOD search method (mixed) to generate conformers. Currently Find Matches always uses the default method, so it will have only one option value.

	Valid values:	default mixed
	Default value:	<b>default</b>
<i>numsteps</i>	An option which sets the number of steps which will be performed during the conformational search. This also limits number of conformations generated.	
	Valid values:	integers
	Default value:	<b>100</b>
	Minimum:	0
<i>postmaxiter</i>	This option determines the maximum number of iterations for post-minimization of generated structures.	
	Valid values:	integers
	Default value:	<b>50</b>
	Minimum:	0
	Maximum:	9999999
<i>postprocessing</i>	Indicates whether or not to perform MacroModel postprocessing.	
	Valid values:	boolean (true false; yes no; y n; on off)
	Default value:	<b>false</b>
<i>premaxiter</i>	This option determines the maximum number of iterations for pre-minimization of input structures.	
	Valid values:	integers
	Default value:	<b>100</b>
	Minimum:	0
	Maximum:	9999999
<i>preprocessing</i>	Indicates whether or not to perform MacroModel preprocessing.	
	Valid values:	boolean (true false; yes no; y n; on off)
	Default value:	<b>false</b>
<i>sampling</i>	This determines whether rapid (standard) or thorough (complete) sampling will be used.	
	Valid values:	standard rapid complete thorough
	Default value:	<b>standard</b>

<i>solvation</i>	This determines whether GB/SA Water (gbsa) or Distance Dependent Dielectric (distance-dependent) solvation treatment is used.
	Valid values: gbsa distance_dependent
	Default value: <b>distance_dependent</b>
<i>window</i>	The energy window ( in kcal/mol ) within which structures will be saved.
	Valid values: reals
	Default value: <b>10</b>
	Minimum: 0.0

## **set**

Creates a new named set. The set name must be a single token (or “quoted” if multiple tokens). A set can be redefined by specifying a new definition.

Syntax:

**set** <set\_name> <ASL-definition>

Operands:

<set\_name> <ASL-definition>

The name which will be applied to the set. If the name contains embedded spaces then it must be enclosed in double quotation marks.

## **setread**

Read set definitions from the file whose name is given as the operand. The file name usually has a “.set” suffix.

Syntax:

**setread** <set\_file\_name>

Operands:

<set\_file\_name>

The name of the file from which the set definitions are to be read. The full name of the file (including any .set suffix) must be specified.

## **setwrite**

Write the currently defined sets to the file whose name is given as the operand. The file name usually has a “.set” suffix

Syntax:

**setwrite** ⟨ set\_file\_name ⟩

Operands:

⟨ set\_file\_name ⟩

The name of the file to which the current set definitions are to be written . The full name of the file (including any .set suffix) must be specified.

## **showhwstereosetup**

Write out the hardware stereo setup.

Syntax:

**showhwstereosetup**

## **showpanel**

Show the panel whose name is given by the operands.

Syntax:

**showpanel** ⟨ panel\_name ⟩ [:⟨ tab\_name ⟩]

Operands:

⟨ panel\_name ⟩ [:⟨ tab\_name ⟩]

The first operand is the name of the panel which is to be displayed. The name must match to all characters. The names of the panels to be used in the “showpanel” command are displayed in parentheses after each item in the main menu bar. The optional second argument is the name of a tab folder within that panel which is to be made the current tab folder. The name of a tab folder to be used in the second optional operand is displayed in the associated panel. The name must match to all characters, but it is not case sensitive.

## **showpanels**

Make visible panels previously hidden with HIDE PANELS

Syntax:

**showpanels**

## **sleep**

This is a standard alias for **energysleep** (see [energysleep], page 101).

## **specifiedname**

Set the specified name to that specified for all atoms which match the ASL specification.

Syntax:

**specifiedname** <specified\_name> <ASL>

Operands:

<specified\_name> <ASL>

The first operand is the specified name which is to be applied to the atom. Only the first 20 characters of the specified name will be used. The second operand is the ASL specification for all the atoms which are to have the specified name applied.

## **spotcenter**

Center the given atom on the screen. Make it the center of global rotation.

Syntax:

**spotcenter** <atom\_num>

Operands:

<atom\_num>

The number of the atom which is to be centered on the screen and to become the center of rotation.

**stop**

This is a standard alias for **energystop** (see [energystop], page 102).

**strikebuildqsar**

Runs a Build QSAR job.

Syntax:

**strikebuildqsar** *activity\_property*=<text> *max\_pls\_factors*=<n>  
    *method*=pls | pca | mlr *remove\_outliers*=yes | no  
    *which\_descriptors*=all | subset <job name>

Options:

*activity\_property*

The name of the activity property.

Valid values: text strings

Default value:

*max\_pls\_factors*

How many factors for Partial Least Squares.

Valid values: integers

Default value: **1**

Minimum: 0

*method* The regression method to use.

Valid values: pls

pca

mlr

Default value: **pls**

*remove\_outliers*

Set to true to automatically remove outliers.

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **false**

*which\_descriptors*

Which descriptors to use.

Valid values: all  
subset  
Default value: **all**

Operands:

$\langle \text{job name} \rangle$

The name of the job to run.

## **strikedeletemodel**

Deletes the given model.

Syntax:

**strikedeletemodel**  $\langle \text{model name} \rangle$

Operands:

$\langle \text{model name} \rangle$

The name of the model to delete.

## **strikeexportmodel**

Exports a model from Strike.

Syntax:

**strikeexportmodel**  $\langle \text{file} \rangle$

Operands:

$\langle \text{file} \rangle$

The file to export the Strike model to.

## **strikeextendselectdescriptor**

Extends the selected descriptors via the given descriptor.

Syntax:

**strikeextendselectdescriptor** < M2IO descriptor name >

Operands:

< M2IO descriptor name >

The M2IO data name of the descriptor to select.

## **strikeimportmodel**

Imports a model to the Strike panels.

Syntax:

**strikeimportmodel** < file >

Operands:

< file >

The file to import the Strike model from.

## **strikepredict**

Runs a prediction job.

Syntax:

**strikepredict** < job name >

Operands:

< job name >

The name of the job to run.

## **strikeselectdescriptor**

Selects the given descriptor

Syntax:

**strikeselectdescriptor** < M2IO descriptor name >

Operands:

< M2IO descriptor name >

The M2IO data name of the descriptor to select.

## **strikeselectmodel**

Selects only the given model.

Syntax:

**strikeselectmodel** < model name >

Operands:

< model name >

The name of the model to select.

## **strikesimilarity**

Runs a similarity job.

Syntax:

**strikesimilarity** *job\_type*=atompairs | descriptors < job name >

Options:

*job\_type* Whether to calculate atom pair or descriptor similarities.

Valid values: atompairs

descriptors

Default value: **atompairs**

Operands:

$\langle \text{job name} \rangle$

The name of the job to run.

### **striketoggleselectdescriptor**

Toggles the selection of the given descriptor on or off.

Syntax:

**striketoggleselectdescriptor**  $\langle \text{M2IO descriptor name} \rangle$

Operands:

$\langle \text{M2IO descriptor name} \rangle$

The M2IO data name of the descriptor to select.

### **structalignatoms**

Sets the ASL that the next struct align job will operate on.

Syntax:

**structalignatoms**  $\langle \text{ASL} \rangle$

Operands:

$\langle \text{ASL} \rangle$

The residues to align.

### **structalignstart**

Start a struct align job with the current settings.

Syntax:

## **structalignstart**

### **substructure**

Specifies a set of atoms to be used as the “substructure” during a substructure minimization or dynamics simulation.

Syntax:

**substructure** *fillres=yes | no* *radius=<x>* *<ASL>*

Options:

*fillres* A boolean option that determines if the substructure definition will be expended to complete residue boundaries.

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **false**

*radius* The radius from the basic substructure definition within which atoms will be included in the substructure.

Valid values: reals  
Default value: **0**  
Minimum: 0.0

Operands:

*<ASL>*

A string in the atom specification language. All atoms which match this will be treated as part of the substructure in a substructure minimization.

### **substructurefile**

This is a standard alias for **substructurefileread** (see [substructurefileread], page 447).

### **substructurefileread**

Will read a .sbc file and replace the current substructure and constrained/fixed atom shells.

Syntax:

**substructurefileread** <sbc\_file\_name>

Operands:

<sbc\_file\_name>

The complete name (including the \*.sbc suffix ) of the file from which the substructure information will be read.

## **substructurefilewrite**

Will write a .sbc file with the current substructure and constrained/fixed atom shells.

Syntax:

**substructurefilewrite** <sbc\_file\_name>

Operands:

<sbc\_file\_name>

The complete name (including the \*.sbc suffix ) of the file to which the substructure information will be written.

Aliases:

**substructurefile** (see [substructurefile], page 446)

## **substructureshell**

Specifies a set of atoms forming a “shell” around the substructure defined by the “subs” command.

Syntax:

**substructureshell** *addatoms*=<text> *constant*=<x>  
  *fillres*=yes | no *frozen*=yes | no *radius*=<x> <shell\_number>

Options:

*addatoms* A string in the atom specification language. All atoms which match this will be treated as part of the shell.

	Valid values:	text strings
	Default value:	
<i>constant</i>	The harmonic force constant to be applied to the constrained shell.	
	Valid values:	reals
	Default value:	<b>200</b>
	Minimum:	0.0
<i>fillres</i>	A boolean option which determines whether the shell is to be made up of complete residues.	
	Valid values:	boolean (true false; yes no; y n; on off)
	Default value:	<b>true</b>
<i>frozen</i>	A boolean option which determines whether the atoms in the shell are to be “frozen” during the substructure calculation.	
	Valid values:	boolean (true false; yes no; y n; on off)
	Default value:	<b>false</b>
<i>radius</i>	The radius of a “shell” of atoms around the “substructure” during a substructure energy procedure.	
	Valid values:	reals
	Default value:	<b>0</b>
	Minimum:	0.0

Operands:

`<shell_number>`

The number of the shell. Shells are usually numbered from 1, but any sequence will work. The shell with the lowest number is defined relative to the atom in the substructure. Subsequent shells are defined relative to next lowest numbered shell.

## superimpose

Perform a superposition using previously defined atoms or all corresponding atoms in all onscreen entries if all is used.

Syntax:

**superimpose** *inplace=yes | no [all]*

Options:

*inplace*    Enable superposition to be performed in place ( effectively just calculates the RMS and doesn't actually move any entries.  
Valid values:    boolean (true|false; yes|no; y|n; on|off)  
Default value:    **false**

Operands:

[all]

If all is present then an attempt will be made to superimpose all onscreen entries. Otherwise only the atom pairs defined by superimposeatom commands will be used to perform a superposition.

## **superimposeatom**

Define an atom pair for which will be superimposed in a subsequent superposition operation.

Syntax:

**superimposeatom** <atom1> <atom2>

Operands:

<atom1> <atom2>

Two atom numbers which represent an atom pair to be superimposed by a subsequent superimpose command. The two atoms must be from different entries and all superimposeatom commands must specify entries in the same order.

## **superimposeset**

Use an ASL expression to define superposition atom pairs. The ASL set must define exactly the same number of atoms in each on-screen entry.

Syntax:

**superimposeset** <ASL>

Operands:

<ASL>

A string in the atom specification language. This set must define exactly the same number of atoms in each on-screen entry and those atoms will become the basis for superposition.

## **superimposesmarts**

Use a SMARTS expression to define superposition atom pairs. The SMARTS expression must define exactly the same number of atoms in each on-screen entry.

Syntax:

**superimposesmarts** <SMARTS>

Operands:

<SMARTS>

A SMARTS expression. This must define exactly the same number of atoms in each on-screen entry and those atoms will become the basis for superposition.

## **surfacedelete**

Deletes the given surface.

Syntax:

**surfacedelete** *entry*=<text> <surface>

Options:

*entry*      The entry that the surface belongs to.

Valid values:    text strings

Default value:

Operands:

<surface>

The name of the surface to delete.

## surfacedisplay

Displays or undisplays the given surface

Syntax:

**surfacedisplay** *display=yes | no entry=<text> <surface>*

Options:

*display* Sets whether or not to display the surface.

Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **true**

*entry* The entry that the surface belongs to.

Valid values: text strings

Default value:

Operands:

*<surface>*

The name of the surface to display.

## surfaceduplicate

Duplicates the given surface.

Syntax:

**surfaceduplicate** *entry=<text> <surface>*

Options:

*entry* The entry that the surface belongs to.

Valid values: text strings

Default value:

Operands:

*<surface>*

The name of the surface to duplicate.

## surfaceextended

Creates a new extended radius surface for the current workspace.

Syntax:

**surfaceextended** *context=entry | molecule | workspace | none | asl* *grid\_spacing=<x> probe\_radius=<x> transparency=<x>*  
name of the surface, followed by ASL defining the atoms to be surfaced

Options:

*context* This option sets the context for generating a surface.

Valid values: entry  
molecule  
workspace  
none  
asl

Default value: **entry**

*grid\_spacing*

This option sets the grid spacing for the surface.

Valid values: reals  
Default value: **0.3**  
Minimum: 0.01

*probe\_radius*

This is the probe radius.

Valid values: reals  
Default value: **1.4**  
Minimum: 1.0

*transparency*

The transparency option controls how transparent the surface appears. A value of 100 means 100 percent transparent. 0 means completely opaque.

Valid values: reals  
Default value: **0**  
Minimum: 0.0  
Maximum: 100.0

Operands:

name of the surface, followed by ASL defining the atoms to be surfaced

Name of the extended radius surface to be created

## **surfaceextendedradiuscontext**

Defines a set of atoms for which a surface can be clipped against with the surfaceextendedradius command.

Syntax:

**surfaceextendedradiuscontext** ⟨ ASL ⟩

Operands:

⟨ ASL ⟩

The ASL expression which defines the atoms the surface will be clipped against.

## **surfaceextendedradiusset**

Defines a set of atoms for which a surface can be created for with the surfaceextendedradius command.

Syntax:

**surfaceextendedradiusset** ⟨ ASL ⟩

Operands:

⟨ ASL ⟩

The ASL expression which defines the atoms the surface will be created for.

## **surfacemolecular**

Creates a new molecular surface for the current workspace.

Syntax:

**surfacemolecular** *context=entry | molecule | workspace | none | asl grid\_spacing=⟨ x ⟩ probe\_radius=⟨ x ⟩ transparency=⟨ x ⟩ ⟨ surface name ⟩*

Options:

*context* This option sets the context for generating a surface.

Valid values: entry  
molecule  
workspace  
none  
asl

Default value: **entry**

*grid\_spacing*

This option sets the grid spacing for the surface.

Valid values: reals  
Default value: **0.8**  
Minimum: 0.01

*probe\_radius*

This is the probe radius.

Valid values: reals  
Default value: **1.4**  
Minimum: 1.0

*transparency*

The transparency option controls how transparent the surface appears. A value of 100 means 100 percent transparent. 0 means completely opaque.

Valid values: reals  
Default value: **0**  
Minimum: 0.0  
Maximum: 100.0

Operands:

*<surface name>*

Name of the molecular surface to be created

## **surfacemolecularcontext**

Defines a set of atoms for which a surface will be clipped against with the surfacemolecular command.

Syntax:

### **surfacemolecularcontext < ASL >**

Operands:

< ASL >

The ASL expression which defines the atoms the surface will be clipped against.

### **surfacemolecularsset**

Defines a set of atoms for which a surface can be created for with the surfacemolecular command.

Syntax:

### **surfacemolecularsset < ASL >**

Operands:

< ASL >

The ASL expression which defines the atoms the surface will be created for.

### **surfacerename**

Renames the given surface.

Syntax:

### **surfacerename entry=< text > newname=< text > < surface >**

Options:

*entry*      The entry that the surface belongs to.

Valid values:    text strings

Default value:

*newname*    The new name for the surface.

Valid values:    text strings

Default value:

Operands:

`<surface>`

The name of the surface to rename.

## **surfacescheme**

Sets the color scheme for the given surface.

Syntax:

```
surfacescheme color=<text> colorramp=<text>
    datarange=on_surface | entire_volume entry=<text>
    negativecolor=<text> scheme=<text> schemevolume=<text>
    <surface>
```

Options:

*color* Sets the color for a constant color.

Valid values: text strings

Default value: **white**

*colorramp* Sets the color ramp for the surface when color scheme is Map Values From Volume.

Valid values: text strings

Default value: **redwhiteblue**

*datarange* Sets the data range to which we apply color ramp.

Valid values: on\_surface

entire\_volume

Default value: **on\_surface**

*entry* The entry that the surface belongs to.

Valid values: text strings

Default value:

*negativecolor*

Sets the color for the negative surface.

Valid values: text strings

Default value: **white**

*scheme* Sets the color scheme.

Valid values: text strings

Default value:

*schemevolume*

Sets the color scheme volume name.

Valid values: text strings

Default value:

Operands:

$\langle \text{surface} \rangle$

The name of the surface to set the color scheme for.

**surfacesetcomment**

Changes the comments of a given surface.

Syntax:

**surfacesetcomment** *comment*= $\langle \text{text} \rangle$  *entry*= $\langle \text{text} \rangle$   $\langle \text{surface} \rangle$

Options:

*comment* The new comments for the surface.

Valid values: text strings

Default value:

*entry* The entry that the surface belongs to.

Valid values: text strings

Default value:

Operands:

$\langle \text{surface} \rangle$

The name of the surface to change the comments for.

**surfacesetisovalue**

Changes the isovalue of a given surface.

Syntax:

### **surfacesetisovalue** *entry*=⟨text⟩ *isovalue*=⟨x⟩ ⟨surface⟩

Options:

*entry* The entry that the surface belongs to.

Valid values: text strings

Default value:

*isovalue* The new isovalue for the surface.

Valid values: reals

Default value: **0.3**

Operands:

⟨surface⟩

The name of the surface to change the isovalue for.

### **surfacesetviewasl**

This command sets a surface to only display the portions of the surface within a given distance of a given ASL.

Syntax:

### **surfacesetviewasl** *distance*=⟨x⟩ *entry*=⟨text⟩ *surface*=⟨text⟩ *use*=yes | no ASL

Options:

*distance* The distance to the ASL.

Valid values: reals

Default value: **5**

*entry* The entry that the surface belongs to.

Valid values: text strings

Default value:

*surface* The name of the surface to operate on.

Valid values: text strings

Default value:

*use* Determines whether or not to use the view by ASL.

Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **true**

Operands:

ASL

The asl which controls the visible surface.

## **surfacestyle**

Sets the drawing style for the given surface.

Syntax:

**surfacestyle** *entry*=⟨text⟩ *style*=solid | mesh | dots ⟨surface⟩

Options:

*entry*      The entry that the surface belongs to.

Valid values:    text strings

Default value:

*style*      Sets the drawing style.

Valid values:    solid

                  mesh

                  dots

Default value:    **solid**

Operands:

⟨surface⟩

The name of the surface to set the style for.

## **surfacetransparency**

Sets the transparency for the given surface.

Syntax:

**surfacetransparency** *entry*=⟨text⟩ *transparency*=⟨x⟩ ⟨surface⟩

Options:

*entry*      The entry that the surface belongs to.

Valid values:    text strings

Default value:

*transparency*

Sets the transparency percentage.

Valid values:    reals

Default value:    **0**

Minimum:        0.0

Maximum:        100.0

Operands:

$\langle \text{surface} \rangle$

The name of the surface to set transparency on.

## surfacevdw

Creates a new vdW surface for the current workspace.

Syntax:

```
surfacevdw asl=⟨text⟩ context=entry | molecule | workspace |
    none | asl grid_spacing=⟨x⟩ transparency=⟨x⟩ Name of the
    surface
```

Options:

*asl*        This option sets the ASL specification for the atoms which define the surface.

Valid values:    text strings  
Default value:

*context*      This option sets the context for generating a surface.

Valid values:    entry  
                  molecule  
                  workspace  
                  none  
                  asl  
Default value:    **entry**

*grid\_spacing*

This option sets the grid spacing for the surface.

Valid values:    reals

Default value:    **0.2**

Minimum:        0.01

*transparency*

The transparency option controls how transparent the surface appears. A value of 100 means 100 percent transparent. 0 means completely opaque.

Valid values:    reals

Default value:    **0**

Minimum:        0.0

Maximum:        100.0

Operands:

Name of the surface

Name of the vdW surface to be created

## **surfacevdwcontext**

Defines a set of atoms for which a surface will be clipped against with the surfacevdw command.

Syntax:

**surfacevdwcontext** < ASL >

Operands:

< ASL >

The ASL expression which defines the atoms the surface will be clipped against.

## **surfacevdwset**

Defines a set of atoms for which a surface can be created for with the surfacevdw command.

Syntax:

**surfacevdwset** < ASL >

Operands:

< ASL >

The ASL expression which defines the atoms the surface will be created for.

## **surfaceviewaslset**

Defines a set of atoms for the View by ASL property which will be used by the surfacesetviewasl command.

Syntax:

**surfaceviewaslset** <ASL>

Operands:

<ASL>

The ASL expression which defines the atoms that will be used for the View by ASL property.

## **symmetrizeworkspace**

Symmetrizes the workspace finding the point groups.

Syntax:

**symmetrizeworkspace** tolerance=<x> analyze|update

Options:

*tolerance*    Tolerance for finding the pointing groups and symmetrizing the workspace

Valid values:    reals

Default value:    **0.04**

Minimum:    0.04

Maximum:    1.00

Operands:

analyze|update

If analyze, finds the point groups for the current tolerance. If update, symmetrize the Workspace with the current tolerance.

## **system**

Execute a command from the system.

Syntax:

**system** ⟨ command ⟩

Operands:

⟨ command ⟩

A command which is to be executed from the current shell.

## **tablealigncolumn**

Set the alignment of the specified column in the current table.

Syntax:

**tablealigncolumn** *alignment*=left | center | right ⟨ columnname ⟩  
⟨ alignment ⟩

Options:

*alignment* The alignment to be set for the column. Valid values are “left”, “center”, or “right”.

Valid values:    left  
                  center  
                  right

Default value: **left**

Operands:

⟨ columnname ⟩ ⟨ alignment ⟩

The name of the column to align. This is the name displayed in the column header.

## **tableresizecolumn**

Set the width of the specified column in the current table.

Syntax:

**tableresizecolumn** < columnname > < width >

Operands:

< columnname > < width >

The name of the column to resize. This is the name displayed in the column header. The width to set the column to. This is the new width for the column.

## **tablesor t**

Sort current entry selection in the specified table.

Syntax:

**tablesor t** *field*=< text > *order*=ascending | descending < table >

Options:

*field*      The name of the property to be sorted on.

Valid values:    text strings

Default value:    **s\_m\_entry\_name**

*order*      This option sets order of sorted values to be either ascending or descending.

Valid values:    ascending

                  descending

Default value:    **ascending**

Operands:

< table >

The name of the table to sort.

## **tablesor tall**

Sort all rows in the specified table.

Syntax:

**tablesortall** <table>

Operands:

<table>

The name of the table to sort.

**tablesortfields**

Sets multiple fields for sorting.

Syntax:

**tablesortfields** <field> <ascending|descending>

Operands:

<field> <ascending|descending>

The fields to sort together with the sort order: ascending or descending.

**tableunselectnonsubset**

Unselects all entries which are not in the current table's subset.

Syntax:

**tableunselectnonsubset**

**tile**

Spread the on-screen entries out in a tile-pattern.

Syntax:

## tile

### toolbarmain

Used to control the main toolbar display features.

Syntax:

**toolbarmain** *location=left | right*

Options:

*location*      Determines main toolbar's location in main drawing window,  
                  1=left and 2=right.

Valid values:      left  
                            right

Default value:      **left**

### torsioncheck

Specifies four atoms which define a torsion to be checked during a conformational search.

Syntax:

**torsioncheck** *maximum=⟨x⟩ minimum=⟨x⟩ ⟨atom1⟩⟨atom2⟩⟨atom3⟩⟨atom4⟩*

Options:

*maximum*      The maximum allowed value for the torsion. during the torsion check.

Valid values:      reals

Default value:      **180**

Minimum:      0.0

*minimum*      The minimum allowed value for the torsion check (Degrees)

Valid values:      reals

Default value:      **0**

Operands:

$\langle \text{atom1} \rangle \langle \text{atom2} \rangle \langle \text{atom3} \rangle \langle \text{atom4} \rangle$

The numbers of four atoms which define a torsion angle to be checked during the conformational search. Note that specifying a-b-c-d is the same as specifying d-c-b-a.

## **torsiongroup**

Sets the current torsion group and the conformation within that group

Syntax:

**torsiongroup**  $\langle \text{group\_name} \rangle \langle \text{conformation\_name} \rangle$

Operands:

$\langle \text{group\_name} \rangle \langle \text{conformation\_name} \rangle$

The first operand must be the name of a torsion group within the current fragment mode. The second operand must be the name of a conformation within that torsion group.

## **transform**

Specify what is to be transformed.

Syntax:

**transform**  $\text{centerlocal=centroid | atom}$   $\text{gui=none | rotate |}$   
 $\text{translate}$   $\text{rsensitivity=}\langle \text{x} \rangle$   $\text{scope=global | local}$   
 $\text{threshold=}\langle \text{n} \rangle$   $\text{tsensitivity=}\langle \text{x} \rangle$  [reset]

Options:

*centerlocal*

Whether local center is an atom or centroid of group of atoms

Valid values:    centroid

                  atom

Default value:    **centroid**

*gui*

Whether rotation, translation or nothing is performed

Valid values: none  
rotate  
translate  
Default value: **rotate**

*rsensitivity*

Mouse rotation sensitivity. Larger is more sensitive.

Valid values: reals  
Default value: **75**  
Minimum: 1.0  
Maximum: 500.0

*scope* Whether global or local transformations are done

Valid values: global  
local  
Default value: **global**

*threshold* How many pixels a mouse move event must contain before any transformation will occur

Valid values: integers  
Default value: **1**  
Minimum: 1  
Maximum: 30

*tsensitivity*

[NOTE: This option is no longer used.] Mouse translation sensitivity. Larger is more sensitive.

Valid values: reals  
Default value: **3.33**  
Minimum: 1.0  
Maximum: 500.0

Operands:

[reset]

Specifies that the transformations are to be reset - this must be reset .

## translate

Translate in Angstroms whatever is specified in the transform set. This is either global (all atoms) or a local grouping defined using an ASL expression in the transform command.

Syntax:

**translate** *x*=⟨*x*⟩ *y*=⟨*x*⟩ *z*=⟨*x*⟩

Options:

*x* Amount in Angstroms to translate in X

Valid values: reals

Default value: **0**

*y* Amount in Angstroms to translate in Y

Valid values: reals

Default value: **0**

*z* Amount in Angstroms to translate in Z

Valid values: reals

Default value: **0**

## **undisplayatom**

Undisplay atoms in the set described by the ASL.

Syntax:

**undisplayatom** ⟨ASL⟩

Operands:

⟨ASL⟩

A string in the atom specification language which describes the set of atoms which are to be undisplayed.

## **undisplaypose**

This command removes the pose set structure from the workspace in the poseview.

Syntax:

**undisplaypose** <pose\_name>

Operands:

<pose\_name>

The name of the pose structure.

**undo**

Undo the effect of the last change on the on-screen structure.

Syntax:

**undo**

**ungroupentries**

Ungroups the entries that match the given ESL expression.

Syntax:

**ungroupentries** <ESL>

Operands:

<ESL>

<ESL> A valid ESL expression to specify which entries are to be ungrouped. Ungroups the entries that match the given ESL expression, and move them all to the end of ungrouped section. In project table, all the ungrouped entries (if any) will be present at the top i.e. before all the groups.

**unhookimport**

Restores normal Import behavior.

Syntax:

**unhookimport**

**uniquepname**

With the set of specified atoms, make the atom name unique. This is done by adding a “~N” to each duplicate name where the “N” is a digit which represents how often this name is repeated in the set.

Syntax:

**uniquepname**

**uniquepdb**

Set unique PDB atom names (by residue) for all atoms that match the ASL specification.

Syntax:

**uniquepdb** <ASL>

Operands:

<ASL>

The operand is a valid ASL string that defines the set of atoms that are to have their PDB atom names changed.

**update**

This is a standard alias for **energyupdate** (see [energyupdate], page 103).

**updateribbons**

Update existing ribbons with current ribbon style and color scheme.

Syntax:

### **updateribbons**

### **varymolecule**

A command which defines a rotatable/translatable molecule during a conformational search. The molecule is defined by specifying any atom which belongs to it.

Syntax:

**varymolecule** *rmax=⟨x⟩ rmin=⟨x⟩ tmax=⟨x⟩ tmin=⟨x⟩*  
⟨atom\_number⟩

Options:

*rmax* The maximum value for the molecule rotation.

Valid values: reals

Default value: **180**

Minimum: 0.0

*rmin* The minimum value for the molecule rotation.

Valid values: reals

Default value: **0**

Minimum: 0.0

*tmax* The maximum value for the molecule translation

Valid values: reals

Default value: **1**

Minimum: 0.0

*tmin* The minimum value for the molecule translation

Valid values: reals

Default value: **0**

Minimum: 0.0

Operands:

⟨atom\_number⟩

The number of an atom which is a member of the molecule which is to be translated or rotated.

**varytorsion**

Defines a rotatable bond to be used in a conformational search.

Syntax:

**varytorsion** *maximum*=⟨x⟩ *minimum*=⟨x⟩ ⟨atom1⟩ ⟨atom2⟩

Options:

*maximum* The maximum value for the torsional rotation.

Valid values: reals

Default value: **180**

Minimum: 0.0

*minimum* The minimum value for the torsional rotation.

Valid values: reals

Default value: **0**

Minimum: 0.0

Operands:

⟨atom1⟩ ⟨atom2⟩

The two atom numbers which define a bond to be rotated in a conformational search. These two atoms must have a bond (usually single) between them. Note that specifying a-b is the same as specifying b-a.

**vcsaddattachment**

Adds an attachment to the core molecule using the given atoms.

Syntax:

**vcsaddattachment** *atom1*=⟨n⟩ *atom2*=⟨n⟩ ⟨attachment name⟩

Options:

*atom1* The atom number of the atom in the original core to set as an attachment point. This is the atom which will be kept.

Valid values: integers

Default value: **1**

Minimum: 1

**atom2**      The atom number of the atom in the original core to set as an attachment point. This is the atom which will be removed.

Valid values:    integers

Default value:    **1**

Minimum:        1

Operands:

**⟨attachment name⟩**

The name of the attachment.

### **vcsaddcorefromproject**

Adds the given entry as a core pose in CombiGlide.

Syntax:

**vcsaddcorefromproject** **⟨entry name⟩**

Operands:

**⟨entry name⟩**

The entry name.

### **vcsaddmincapcore**

Adds the minimally capped core to the poses for Define Core Poses in CombiGlide.

Syntax:

**vcsaddmincapcore**

### **vcsaddoriginalcore**

Adds the original core to the poses for Define Core Poses in CombiGlide.

Syntax:

**vcsaddoriginalcore**

**vcscanceldockjob**

Cancels the currently running Dock Library job associated results.

Syntax:

**vcscanceldockjob**

**vcsclearreagentfile**

Clears the reagent file for the selected rows.

Syntax:

**vcsclearreagentfile**

**vcscombiexportdockingfile**

Exports the combinatorial docking results from CombiGlide to the given file.

Syntax:

**vcscombiexportdockingfile** <file name>

Operands:

<file name>

The file name.

**vcscombiexportdockingproject**

Exports the combinatorial docking results from CombiGlide.

Syntax:

### **vcscombiexportdockingproject**

### **vcscombiexportoptions**

Holds the options for exporting combinatorial docking results from CombiGlide.

Syntax:

**vcscombiexportoptions** *includereceptor=yes | no*  
*numreagents=<n>*

Options:

*includereceptor*

Indicates whether or not to include the receptor in the exported results

Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **false**

*numreagents*

The number of reagents to export

Valid values: integers

Default value: **100**

Minimum: 1

### **vcsconfiguredocking**

Allows the settings of some values that determine how the overall VCS job runs.

Syntax:

```
vcsconfigureddocking gridfilename=⟨text⟩ lig_ccut=⟨x⟩
    lig_vscale=⟨x⟩ ligandwithmetal=charged | either | neutral
    maxatom=⟨n⟩ maxrotbonds=⟨n⟩ numreqgroup1=⟨n⟩
    numreqgroup2=⟨n⟩ numreqgroup3=⟨n⟩ numreqgroup4=⟨n⟩
    penalizeamidebondrotations=yes | no reqmodegroup1=all |
    atleast reqmodegroup2=all | atleast reqmodegroup3=all |
    atleast reqmodegroup4=all | atleast ringconf=yes | no
```

Options:

*gridfilename*

The base name for the file which the receptor grid is to be written to or read from.

Valid values: text strings

Default value:

*lig\_ccut* The partial atomic charge below which ligand atoms are considered to be non-polar and will have their VDW radii scaled.

Valid values: reals

Default value: **0.15**

Minimum: 0.00000001

*lig\_vscale* The scaling factor for the VDW radii of non-polar ligand atoms.

Valid values: reals

Default value: **0.8**

Minimum: 0.00000001

*ligandwithmetal*

Controls which ligand atoms can interact with metal sites.

Valid values: charged

either

neutral

Default value: **charged**

*maxatom* Any ligands in the input with more than this number of atoms will be skipped.

Valid values: integers

Default value: **120**

Minimum: 1

Maximum: 200

*maxrotbonds*

Any ligands in the input with more than this number of rotatable bonds will be skipped.

Valid values: integers

Default value: **20**

Minimum: 1

*numreqgroup1*

Number of constraints to be required for group 1 in docking.

Valid values: integers

Default value: **1**

Minimum: 0

Maximum: 4

*numreqgroup2*

Number of constraints to be required for group 2 in docking.

Valid values: integers

Default value: **1**

Minimum: 0

Maximum: 4

*numreqgroup3*

Number of constraints to be required for group 3 in docking.

Valid values: integers

Default value: **1**

Minimum: 0

Maximum: 4

*numreqgroup4*

Number of constraints to be required for group 4 in docking.

Valid values: integers

Default value: **1**

Minimum: 0

Maximum: 4

*penalizeamidebondrotations*

An option that penalizes twisted (non-planar) amide bonds.

Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **true**

*reqmodegroup1*

The mode determines how to set the number of required constraints for group 1 in docking.

Valid values: all

atleast

Default value: **atleast**

*reqmodegroup2*

The mode determines how to set the number of required constraints for group 2 in docking.

Valid values: all

atleast

Default value: **atleast**

***reqmodegroup3***

The mode determines how to set the number of required constraints for group 3 in docking.

Valid values:    all  
                  atleast  
Default value: **atleast**

***reqmodegroup4***

The mode determines how to set the number of required constraints for group 4 in docking.

Valid values:    all  
                  atleast  
Default value: **atleast**

***ringconf*** An option which allows ring flips

Valid values:    boolean (true|false; yes|no; y|n; on|off)  
Default value: **true**

**vcscoreoptions**

Sets the options for Define Core Poses.

Syntax:

**vcscoreoptions** *centroidx=⟨ x ⟩ centroidy=⟨ x ⟩ centroidz=⟨ x ⟩*  
*constrainradius=⟨ x ⟩ maxrmsd=⟨ x ⟩ poseconstraint=box |*  
*sphere | position*

Options:

*centroidx* The X-coordinate of the location to constrain the core center of mass to.

Valid values:    reals  
Default value: **0**

*centroidy* The Y-coordinate of the location to constrain the core center of mass to.

Valid values:    reals  
Default value: **0**

*centroidz* The Z-coordinate of the location to constrain the core center of mass to.

Valid values:    reals  
Default value: **0**

*constrainradius*

The radius of the constraint for the core's center of mass.

Valid values:    reals

Default value:    **5**

Minimum:        1.0

*maxrmsd*    The maximum RMSD that the core can move.

Valid values:    reals

Default value:    **2**

Minimum:        0.0

*poseconstraint*

Controls the allowed placement for core structures in CombiGlide docking.

Valid values:    box

sphere

position

Default value:    **box**

## **vcscreatedockedlibrary**

Start the creation and docking of the library in the Analyse Library step of CombiGlide. This is similar to vcsrunenumerateddocking, except that it uses only the selected reagents at each position and does not remove or replace contents of the Combinatorial Screening run.

Syntax:

## **vcscreatedockedlibrary**

## **vcsdeleteattachment**

Deletes all of the selected attachments.

Syntax:

**vcsdeleteattachment**

**vcsdeletecore**

Deletes the selected cores in the Define Core Poses step in CombiGlide.

Syntax:

**vcsdeletecore**

**vcsdeleteresults**

Deletes the given results.

Syntax:

**vcsdeleteresults** <name>

Operands:

<name>

The name of the results file.

**vcsdisplayreceptor**

This function displays the receptor for the current CombiGlide run in the Workspace.

Syntax:

**vcsdisplayreceptor**

**vcsenumeratedockoptions**

Holds the options for enumerate and dock for CombiGlide.

Syntax:

**vcsenumeratedockoptions** *mode*=⟨ text ⟩

Options:

*mode* An option controlling what type of docking will be done after combinatorial enumeration. The allowed values are **combi** , **xp** , **sp** , and **htvs** .

Valid values: text strings

Default value: **combi**

## **vcsexcludetablerow**

Excludes the given row in the structure table in the step from the Workspace.

Syntax:

**vcsexcludetablerow** ⟨ row ⟩

Operands:

⟨ row ⟩

The row number to exclude in the Workspace.

## **vcsexportdefinition**

Stores the current core molecule and attachments in a file.

Syntax:

**vcsexportdefinition** ⟨ file name ⟩

Operands:

⟨ file name ⟩

The name of the file to store the core definition in.

## **vcsexportresults**

Stores the settings from the Filter and Select dialog, and all the results, in a human-readable text file.

Syntax:

**vcsexportresults** <file name>

Operands:

<file name>

The name of the file to store the settings and results in.

## **vcsimportdefinition**

Reads a core molecule and attachments from the given file.

Syntax:

**vcsimportdefinition** <file name>

Operands:

<file name>

The name of the file to read the core definition from.

## **vcsincludeextendtablerow**

Extends the rows included in the workspace to include this one.

Syntax:

**vcsincludeextendtablerow** <row>

Operands:

<row>

The row number to include in the Workspace.

### **vcsincludeonlytablerow**

Includes only the given row in the structure table in the step into the Workspace.

Syntax:

**vcsincludeonlytablerow** <row>

Operands:

<row>

The row number to include in the Workspace.

### **vcsincludetablerow**

Includes the given row in the structure table in the step into the Workspace.

Syntax:

**vcsincludetablerow** <row>

Operands:

<row>

The row number to include in the Workspace.

### **vcsinverttableselection**

Inverts the row selection in the first table in the step.

Syntax:

**vcsinverttableselection**

### **vcsoptions**

This command holds general options for CombiGlide.

Syntax:

**vcsoptions** *mode*=⟨text⟩ *sidechainnode*=⟨n⟩ *untangle*=yes | no

Options:

*mode* An option controlling what type of job CombiGlide will run.

Valid values: text strings

Default value: **sidechain**

*sidechainnode*

Which sidechain node we are working on.

Valid values: integers

Default value: **1**

Minimum: 1

*untangle* An option which allows post-combgen minimization (for Create Library, enumeration only).

Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **true**

## vcsrefreshstructure

This function refreshes the structure in the Workspace from the current core structure in CombiGlide.

Syntax:

**vcsrefreshstructure** *viewcappedcore*=yes | no

Options:

*viewcappedcore*

An option which allows viewing of the minimally capped core, rather than the original core, in the Workspace.

Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **false**

## vcsrenameattachment

Renames the attachment in CombiGlide to the new name.

Syntax:

**vcsrenameattachment** *row=⟨ n ⟩ ⟨ new name ⟩*

Options:

*row*      The row to rename.  
Valid values:    integers  
Default value:    **1**  
Minimum:        1

Operands:

⟨ new name ⟩

The new name for the attachment.

## **vcsrestoreresults**

Restores the filter and selection settings, and all associated results.

Syntax:

**vcsrestoreresults** ⟨ name ⟩

Operands:

⟨ name ⟩

The name of the results file.

## **vcsruncombinatorialdocking**

Runs a combinatorial docking job for CombiGlide.

Syntax:

**vcsruncombinatorialdocking** *maxresults=⟨ n ⟩ ⟨ job name ⟩*

Options:

*maxresults*      The number of combinatorial results to return.

Valid values: integers  
Default value: **1000**  
Minimum: 0

Operands:  
 $\langle \text{job name} \rangle$   
The job name.

## **vcsruncombinatorialselection**

Runs a combinatorial selection job for CombiGlide.

Syntax:

**vcsruncombinatorialselection** *cmdargs*= $\langle \text{text} \rangle$

Options:

*cmdargs* The arguments for the reagent selection command.  
Valid values: text strings  
Default value:

## **vcsruncreate**

Creates the run with the given name.

Syntax:

**vcsruncreate**  $\langle \text{run name} \rangle$

Operands:  
 $\langle \text{run name} \rangle$   
The name of the new run to create.

## **vcsrundelete**

Deletes the current run from the project.

Syntax:

**vcsrundelete**

**vcsrunenumerateddock**

Runs an enumerated docking job in the Dock Library step of CombiGlide.

Syntax:

**vcsrunenumerateddock**

**vcsrunopen**

Opens the run with the given name.

Syntax:

**vcsrunopen** <run name>

Operands:

<run name>

The name of the run to open.

**vcsrunrename**

Changes the current run's name to the given name.

Syntax:

**vcsrunrename** <run name>

Operands:

<run name>

The name to change the current run's name to.

### **vcsruncsaveas**

Saves a copy of the current run under the given name.

Syntax:

**vcsruncsaveas** ⟨run name⟩

Operands:

⟨run name⟩

The name of the run to save as.

### **vcsruncsingledocking**

Runs a single position docking job for CombiGlide.

Syntax:

**vcsruncsingledocking**

### **vcsruncsingleselection**

Runs a single position selection job for CombiGlide.

Syntax:

**vcsruncsingleselection** cmdargs=⟨text⟩

Options:

cmdargs      The arguments for the reagent selection command.

Valid values:    text strings

Default value:

### **vcssaveresults**

Saves the current filter and selection settings, and all associated results.

Syntax:

**vcssaveresults** <name>

Operands:

<name>

The name of the results file.

### **vcsselectalltablerows**

Selects all rows in the first table in the step.

Syntax:

**vcsselectalltablerows**

### **vcsselectextendtablerow**

Extends the selection to this row in the table.

Syntax:

**vcsselectextendtablerow** <row>

Operands:

<row>

The row number to extend the select to.

### **vcsselectonlytablerow**

Selects only this row in the table.

Syntax:

**vcsselectonlytablerow** <row>

Operands:

<row>

The row number to select only in the table row.

**vcsselecttablerow**

Selects the given row in the first table in the step.

Syntax:

**vcsselecttablerow** <row>

Operands:

<row>

The row number to select in the table.

**vcssetattachmentfile**

Sets the reagent file for the given attachment.

Syntax:

**vcssetattachmentfile** *file*=<text> <attachment name>

Options:

*file*      The file name of the reagent file to add to the given attachment.

Valid values:    text strings

Default value:

Operands:

<attachment name>

The name of the attachment.

## **vcssetmolecule**

Sets the core molecule for the current CombiGlide run to the molecule containing the given atom.

Syntax:

**vcssetmolecule** *title*=⟨text⟩ ⟨atom number⟩

Options:

*title* This option sets the title for the core molecule.

Valid values: text strings

Default value: **core**

Operands:

⟨atom number⟩

The atom number of the molecule.

## **vcssetreagentfile**

Sets the reagent file for the selected rows.

Syntax:

**vcssetreagentfile** ⟨reagent name⟩

Operands:

⟨reagent name⟩

The name of the reagent file.

## **vcssingleexportdockingfile**

Exports the single-position docking results from CombiGlide to the given file.

Syntax:

### **vcssingleexportdockingfile** <file name>

Operands:

<file name>

The file name.

### **vcssingleexportdockingproject**

Exports the single-position docking results from CombiGlide.

Syntax:

### **vcssingleexportdockingproject**

### **vcssingleexportoptions**

Holds the options for exporting single-position docking results from CombiGlide.

Syntax:

**vcssingleexportoptions** *includereceptor=yes | no*  
*numreagents=<n>*

Options:

#### *includereceptor*

Indicates whether or not to include the receptor in the exported results

Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **false**

#### *numreagents*

The number of reagents to export

Valid values: integers

Default value: **100**

Minimum: 1

## **vcssorttable**

Resort the given CombiGlide table based on the data in the specified column

Syntax:

**vcssorttable** *table=⟨ n ⟩ ⟨ column\_name ⟩*

Options:

*table*      The table to set.

Valid values:    integers

Default value:    **10**

Operands:

⟨ column\_name ⟩

The name of the column to be sorted.

## **vcsstepforward**

Moves forward to the next VCS step. Deletes any steps after the current step, then creates the next step, using the data from previous steps.

Syntax:

**vcsstepforward**

## **vcsstepgoto**

Moves to an existing step in the current project.

Syntax:

**vcsstepgoto** ⟨ step name ⟩

Operands:

⟨ step name ⟩

The name of the step to switch to.

## **vcsunselectablerow**

Unselects the given row in the first table in the step.

Syntax:

**vcsunselectablerow** <row>

Operands:

<row>

The row number to unselect in the table.

## **viewmatrix**

Sets workspace rotation matrix to the matrix supplied When 'inverse' is supplied, it sets the inverse roation matrix and when 'nocenter', No center rotation matrix. And in the absence of both the keywords, it sets the rotation matrix.

Syntax:

**viewmatrix**

## **viewposecontacts**

This command alters contact settings, to display bad and ugly receptor/ligand contacts in the pose viewer.

Syntax:

**viewposecontacts**

## **viewposebonds**

This command alters hbond settings, to display receptor/ligand H-bonds in the pose viewer.

Syntax:

**viewposehbonds**

**viewreset**

Resets the viewing transform to original.

Syntax:

**viewreset**

**viewrestore**

Sets the current viewing transform to be the saved or “home” transform.

Syntax:

**viewrestore**

**viewsave**

Saves the current viewing transform.

Syntax:

**viewsave**

**viewvolume**

Sets workspace view bounding box to the supplied values

Syntax:

**viewvolume** *bottom*=⟨x⟩ *far*=⟨x⟩ *left*=⟨x⟩ *near*=⟨x⟩  
  *right*=⟨x⟩ *top*=⟨x⟩

Options:

*bottom*      Bottom co-ordinate of the bounding box

Valid values:    reals

Default value: **-5**

*far*        Far co-ordinate of the bounding box

Valid values:    reals

Default value: **-10**

*left*       Left co-ordinate of the bounding box

Valid values:    reals

Default value: **-5**

*near*       Near co-ordinate of the bounding box

Valid values:    reals

Default value: **10**

*right*       Right co-ordinate of the bounding box

Valid values:    reals

Default value: **5**

*top*        Top co-ordinate of the bounding box

Valid values:    reals

Default value: **5**

## visimport

Creates all volumes and surfaces in the given file.

Syntax:

**visimport** *entry*=⟨text⟩ *isovalue*=⟨x⟩ *transparency*=⟨x⟩  
  ⟨file.vis⟩

Options:

*entry*       This is the entry to associate the objects with.

Valid values:    text strings

Default value:

*isovalue* If any of the volumes do not have associated surfaces or suggested isovalue, then this isovalue will be used to isosurface the volumes.

Valid values: reals  
Default value: **0.03**

*transparency*

This option sets the transparency of any surfaces.

Valid values: reals  
Default value: **0**  
Minimum: 0.0  
Maximum: 100.0

Operands:

$\langle \text{file.vis} \rangle$

The name of the visio or Jaguar plot file to load.

## **volumedelete**

Deletes the given volume.

Syntax:

**volumedelete** *entry*= $\langle \text{text} \rangle$   $\langle \text{volume} \rangle$

Options:

*entry* The entry that the volume belongs to.  
Valid values: text strings  
Default value:

Operands:

$\langle \text{volume} \rangle$

The name of the volume to delete.

## **volumeisosurface**

Isosurfaces the given volume.

Syntax:

**volumeisosurface** *entry*=⟨text⟩ *isovalue*=⟨x⟩ *surface*=⟨text⟩  
⟨volume⟩

Options:

*entry*      The entry that the volume belongs to.

Valid values:    text strings

Default value:

*isovalue*    The isovalue to use for the isosurfacing.

Valid values:    reals

Default value: **0.1**

*surface*     The name for the new surface.

Valid values:    text strings

Default value:

Operands:

⟨volume⟩

The name of the volume to isosurface.

## volumerename

Renames the given volume.

Syntax:

**volumerename** *entry*=⟨text⟩ *newname*=⟨text⟩ ⟨volume⟩

Options:

*entry*      The entry that the volume belongs to.

Valid values:    text strings

Default value:

*newname*    The new name for the volume.

Valid values:    text strings

Default value:

Operands:

`< volume >`

The name of the volume to rename.

## wake

This is a standard alias for `energywake` (see [energywake], page 104).

## workspaceselectionadd

Extends workspace selection with the atoms confirming to the supplied ASL.

Syntax:

`workspaceselectionadd < ASL >`

Operands:

`< ASL >`

ASL representing the set of atoms

## workspaceselectionclear

Resets current workspace selection to none.

Syntax:

`workspaceselectionclear`

## workspaceselectioninvert

Flips the state of an atom in the workspace selection. i.e., removes if the atom already exists or adds if it doesn't.

Syntax:

### **workspaceselectioninvert <ASL>**

Operands:

<ASL>

ASL representing the set of atoms

### **workspaceselectionreplace**

Overwrites the workspace selection with the supplied set of atoms.

Syntax:

### **workspaceselectionreplace <ASL>**

Operands:

<ASL>

ASL representing the set of atoms

### **workspaceselectionsubtract**

Removes atoms confirming to the supplied ASL from the workspace selection.

Syntax:

### **workspaceselectionsubtract <ASL>**

Operands:

<ASL>

ASL representing the set of atoms

### **write**

This is a standard alias for **filewrite** (see [filewrite], page 140).

## **writeangle**

Write the angles in the angle table to a file.

Syntax:

```
writeangle atomnumber=entry | molecule | workspace
    delimiter=comma | tab | userdefined userdelimiter=<text>
    <file_name>
```

Options:

*atomnumber*

This option is deprecated.

Valid values: entry  
molecule  
workspace

Default value: **entry**

*delimiter* This option sets what delimiter will be used in the output file:  
comma=1, tab=2, and userdefined=3.

Valid values: comma  
tab  
userdefined

Default value: **comma**

*userdelimiter*

The delimiter string defined by users. It is valid only when option delimiter is userdefined.

Valid values: text strings

Default value:

Operands:

<file\_name>

The name of the file to which the angles will be written.

## **writecontact**

Write Contacts information to a file.

Syntax:

**writecontact** *atomnumber=entry | molecule | workspace*  
    *delimiter=comma | tab | userdefined userdelimiter=<text>*  
    *<file\_name>*

Options:

*atomnumber*

This option is deprecated.

Valid values: entry  
                 molecule  
                 workspace

Default value: **entry**

*delimiter* This option sets what delimiter will be used in the output file:  
comma=1, tab=2, and userdefined=3.

Valid values: comma  
                 tab  
                 userdefined

Default value: **comma**

*userdelimiter*

The delimiter string defined by users. It is valid only when option delimiter is userdefined.

Valid values: text strings

Default value:

Operands:

*<file\_name>*

The name of the file to which Contacts information will be written.

## **writecoupling**

Write the coupling constants in the coupling table to a file.

Syntax:

**writecoupling** *atomnumber=entry | molecule | workspace*  
    *delimiter=comma | tab | userdefined userdelimiter=<text>*  
    *<file\_name>*

Options:

*atomnumber*

This option is deprecated.

Valid values: entry  
molecule  
workspace  
Default value: **entry**

*delimiter* This option sets what delimiter will be used in the output file:  
comma=1, tab=2, and userdefined=3.  
Valid values: comma  
tab  
userdefined  
Default value: **comma**

*userdelimiter*  
The delimiter string defined by users. It is valid only when option delimiter is userdefined.  
Valid values: text strings  
Default value:

Operands:  
*<file\_name>*  
The name of the file to which the coupling constants will be written.

## writedihedral

Write the dihedrals in the dihedral table to a file.

Syntax:

```
writedihedral atomnumber=entry | molecule | workspace  
           delimiter=comma | tab | userdefined userdelimiter=<text>  
           <file_name>
```

Options:

*atomnumber*

This option has been deprecated.

Valid values: entry  
molecule  
workspace  
Default value: **entry**

*delimiter* This option sets what delimiter will be used in the output file:  
comma=1, tab=2, and userdefined=3.

Valid values: comma  
tab  
userdefined  
Default value: **comma**

*userdelimiter*

The delimiter string defined by users. It is valid only when option delimiter is userdefined.

Valid values: text strings  
Default value:

Operands:

*<file\_name>*

The name of the file to which the dihedrals will be written.

## writedistance

Write the distances in the distance table to a file.

Syntax:

```
writedistance atomnumber=entry | molecule | workspace  
    delimiter=comma | tab | userdefined userdelimiter=<text>  
    <file_name>
```

Options:

*atomnumber*

This option is deprecated.

Valid values: entry  
molecule  
workspace  
Default value: **entry**

*delimiter*

This option sets what delimiter will be used in the output file:  
comma=1, tab=2, and userdefined=3.

Valid values: comma  
tab  
userdefined  
Default value: **comma**

*userdelimiter*

The delimiter string defined by users. It is valid only when option delimiter is userdefined.

Valid values: text strings  
Default value:

Operands:

$\langle \text{file\_name} \rangle$

The name of the file to which the distances will be written.

## **writehbond**

Write H-Bonds information to a file.

Syntax:

**writehbond** *atomnumber=entry | molecule | workspace*  
*delimiter=comma | tab | userdefined userdelimiter=( text )*  
 $\langle \text{file\_name} \rangle$

Options:

*atomnumber*

This option is deprecated.

Valid values: entry  
molecule  
workspace

Default value: **entry**

*delimiter* This option sets what delimiter will be used in the output file:  
comma=1, tab=2, and userdefined=3.

Valid values: comma  
tab  
userdefined

Default value: **comma**

*userdelimiter*

The delimiter string defined by users. It is valid only when option delimiter is userdefined.

Valid values: text strings  
Default value:

Operands:

$\langle \text{file\_name} \rangle$

The name of the file to which H-Bonds information will be written.

## **writeposefile**

If the write command is issued with options, but without a filename then nothing is written but the options are updated. If the write command alone is issued then writing is performed to the currently open file with the current options. A write command with both options and a file name specified will result in the write being performed with the new options to the specified file.

Syntax:

**writeposefile** *append=yes | no <file\_name>*

Options:

*append* This option determines whether to append to the file which is going to be written.

Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **false**

Operands:

*<file\_name>*

The name of the file to which the structure will be written from the pose-viewer.

## **xcluster**

This keyword is used to set various options associated with starting XCluster jobs from Maestro.

Syntax:

**xcluster** *cluster\_by=atom | torsion compare\_enantiomers=yes | no  
input\_file=<text> job=<text> rms\_in\_place=yes | no  
structure\_source=selected\_entries | workspace | file*

Options:

*cluster\_by* Whether to use atomic RMS or torsional RMS differences for clustering analysis.

Valid values: atom  
torsion

Default value: **atom**

*compare\_enantiomers*

A boolean which controls whether XCluster will perform superposition in such a way so as to compare enantiomers.

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **true**

*input\_file* The name of the structure input file.

Valid values: text strings  
Default value:

*job* The name for the XCluster job.

Valid values: text strings  
Default value: **xclustmp**

*rms\_in\_place*

A boolean which controls whether XCluster will calculate RMS differences without first doing a superposition.

Valid values: boolean (true|false; yes|no; y|n; on|off)  
Default value: **false**

*structure\_source*

Whether to use the selected entries in the current project, or a specified file with multiple structures as structure input for the job.

Valid values: selected\_entries  
workspace  
file

Default value: **selected\_entries**

## **xclusterstart**

Start a XCluster job with the current settings.

Syntax:

## **xclusterstart**

## **xclusterwrite**

Write a XCluster input file with the current settings.

Syntax:

**xclusterwrite**

**zoom**

Zoom in or out (i.e. scale up or down)

Syntax:

**zoom** *factor=⟨x⟩ in | out*

Options:

*factor* A zooming (scaling) factor which determines by how much the on-screen structures is scaled. 1.0 is 1 percent, 99.0 is 99 percent, etc. Values are used relative to current size

Valid values:    reals

Default value:    **10**

Minimum:    1.0

Maximum:    99.0

Operands:

in | out

Must specify whether to zoom in and make larger or out and make smaller.



# Command Index

## 1

1ddataset.....	29
1dplot.....	31
1drescale.....	32
1dtable.....	32

## 2

2ddataset.....	32
2dplot.....	33
2drescale.....	35

## A

addfromhold .....	35
adjustangle .....	35
adjustdihedral .....	36
adjustdistance .....	37
alias .....	38
angle .....	38
appendribbons .....	38
application .....	39
atom .....	39
atomname .....	40
attach .....	40
attachmentmarkerdump .....	41
attachmentmarkersettings .....	41
autosetup .....	44

## B

beginundoblock .....	45
bmincomfile .....	45
bond .....	46
bondorder .....	46
bondtonew .....	46
buildoptions .....	47

## C

calcenergy .....	47
canonicalname .....	48
cascadepanels .....	48
cellsmarkerdump .....	49
cellsmarkersettings .....	49
centeratom .....	52
centerbond .....	53

centercoordinates .....	53
centroid .....	54
centroidatom .....	54
cglidedockconstraintposition .....	54
cglidedockconstraintregion .....	56
chainname .....	57
changedirectory .....	57
chiralatom .....	58
clip .....	58
clusteratom .....	59
clusterheavyatoms .....	59
clusterset .....	59
clustertorsion .....	60
coloratom .....	60
colorscheme .....	61
combilibenum .....	61
combilibenumaddattachment .....	61
combilibenumclearreagentfile .....	62
combilibenumdeleteattachment .....	62
combilibenumexportdefinition .....	63
combilibenumimportdefinition .....	63
combilibenumoptions .....	63
combilibenumrefreshstructure .....	64
combilibenumrenameattachment .....	64
combilibenumselectextendtablerow .....	65
combilibenumselectonlytablerow .....	65
combilibenumselecttablerow .....	65
combilibenumsetmolecule .....	66
combilibenumsetreagentfile .....	66
combilibenumunselecttablerow .....	66
compareatom .....	67
compareset .....	67
confelim .....	68
confelimstart .....	69
confelimwrite .....	69
confgenltsearch .....	69
confgemini .....	72
confgenpotential .....	73
confgenreadpotential .....	75
confgenstart .....	75
confgenwrite .....	75
confsearch .....	76
connect .....	78
connectfuseatom .....	78
constrainedangle .....	79
constrainedatom .....	79
constraineddist .....	80

constrainedset .....	81	entryextendselction .....	111
constrainedtorsion .....	82	entryextendwsinclude .....	111
contactcriteria .....	82	entrygroupcollapse .....	112
contactset1 .....	83	entrygroupcreate .....	112
contactset2 .....	84	entrygroupdelete .....	112
coupling .....	84	entrygroupduplicate .....	113
createlibrary .....	85	entrygroupexpand .....	113
createpropsubset .....	85	entrygroupexpandonly .....	114
createsubset .....	85	entrygroupextendselection .....	114
<b>D</b>			
defaultfc .....	86	entrygroupmove .....	114
delete .....	86	entrygrouprename .....	115
dialogoptions .....	87	entrygroupselect .....	115
dihedral .....	87	entrygroupselectonly .....	115
dihedraldrive .....	88	entrygroupsettitle .....	116
displayatom .....	88	entrygroupungroup .....	116
displayonlyatom .....	89	entrygroupunselect .....	117
displayonlypose .....	89	entrygroupwsexclude .....	117
displayopt .....	89	entrygroupwsinclude .....	117
displaypose .....	93	entrygroupwsincludeonly .....	118
distance .....	93	entryimport .....	118
distancecheck .....	94	entryimportsspreadsheet .....	120
dynamics .....	94	entryimportvibration .....	121
<b>E</b>			
ejob .....	96	entryinvertselection .....	122
elementlabels .....	97	entrymovetogroup .....	122
embrace .....	97	entryrename .....	122
embracecsearch .....	98	entryselect .....	123
endundoblock .....	100	entryselectall .....	123
energykill .....	100	entryselectonly .....	123
energymonitor .....	101	entryselectonlyrow .....	124
energysleep .....	101	entryselectrandom .....	124
energystart .....	102	entryselectrow .....	125
energystop .....	102	entrysetprop .....	125
energytask .....	103	entrysettitle .....	126
energyupdate .....	103	entryshowall .....	126
energyupdatejobstatus .....	103	entrytable .....	126
energywake .....	104	entryunselect .....	127
enhance3d .....	104	entryunselectall .....	127
entrycombine .....	105	entryunselectrow .....	127
entrycopyprop .....	105	entrywscreate .....	128
entrydelete .....	106	entrywsexclude .....	128
entrydisassociate .....	107	entrywsexcludenotfixed .....	128
entrydragselection .....	107	entrywsinclude .....	129
entryduplicate .....	107	entrywsincludelock .....	129
entryexport .....	108	entrywsincludeonly .....	129
entryexportsspreadsheet .....	110	epik .....	130
		epikstart .....	132
		eplayergoto .....	132
		eplayergotofirst .....	132
		eplayergotolast .....	133
		eplayersettings .....	133
		eplayerstepahead .....	135

eplayerstepback.....	135	glideligand.....	164
errorcheck.....	135	glideligandfile.....	166
excludedvolumesmarkersettings....	136	glideligandgrid.....	166
extenddisplaytopose.....	137	glidelocatehydrophobiccells.....	168
		glideoutput.....	168
		glidereceptorligand.....	169
		gliderefenceligand.....	169
		glidescoring.....	170
fileread.....	138	glidesettings.....	172
filewrite .....	140	glidesite.....	175
filter.....	142	grow.....	177
find.....	142	growbond.....	177
fit.....	145	growbond2.....	178
forcefield.....	145	growdirection.....	178
forcefieldbend.....	145	growname.....	178
forcefielddele.....	146		
forcefieldgbsolv.....	147		
forcefieldimproper.....	147		
forcefieldsasolv.....	148		
forcefieldstretch.....	149		
forcefieldtorsion.....	150		
forcefieldvdw.....	150		
forcefieldview.....	151		
forcefieldwilson.....	151		
formalcharge.....	152		
fragment.....	153		
fragmenttype.....	153		
frozenatom.....	153		
frozensest.....	154		
fuse.....	154		
<b>G</b>			
geometrycleanup.....	154	hbondcriteria.....	179
glideactivesiteres.....	155	hbondset1.....	180
glideactivesiteset.....	155	hbondset2.....	180
glidecalcboxfromligand.....	155	helpauto.....	181
glideconstraintatomlabel.....	156	helpcategory.....	181
glideconstraintposition.....	156	helpsearch.....	182
glideconstraintregion.....	157	helptopic.....	182
glideconstraintregionactivecell..	157	hideentries.....	183
glideconstraintregiongrow.....	157	hidepanel.....	183
glideconstraintregioninvisible...	158	hidepanels.....	183
glideconstraintregionshrink.....	158	hideproperty.....	184
glideconstraintregionvisible....	158	historyvisible.....	184
glidedisplayreceptor.....	159	hold.....	191
glidedockconstraintposition.....	159	hppmap.....	191
glidedockconstraintregion.....	160	hppmapbox.....	192
glidedockregionnumatoms.....	161	hppmapset.....	193
glidedockregionselect.....	162	hppmapstart.....	193
glidedockregionunselect.....	162	hppmapwrite.....	193
glidegridhydrophobic.....	163	hydrogenapply.....	194
		hydrogentreat.....	194
<b>I</b>			
		impactbufferedatom.....	195
		impactbufferedset.....	195
		impactconstraints.....	195
		impactcontinuumsolvent.....	196
		impactdynamics.....	197
		impactfastmultipole.....	199
		impactfrozenatom.....	200
		impactfrozensest.....	200
		impacthybridmc.....	201
		impactjob.....	201

impactmdparams .....	202	mcsd .....	239	
impactminimization .....	204	minienergy .....	240	
impactperiodicboundary .....	206	minta .....	241	
impactpotential .....	207	monitorangle .....	242	
impactstart .....	209	monitordistance .....	242	
impacttask .....	209	monitorhbond .....	243	
impacttruncation .....	209	monitorsetsurf .....	244	
impactwrite .....	210	monitorsettings .....	244	
invert .....	210	monitorsurf .....	245	
invertset .....	210	monitortorsion .....	245	
<b>J</b>				
jaguarassignatomnames .....	211	mouse .....	245	
jaguarimportgeometry .....	211	move .....	246	
jaguarinputfilesjob .....	212	multiplemini .....	246	
jaguarjob .....	213	mutate .....	247	
jaguarselectextendtablerow .....	213	<b>N</b>		
jaguarselectonlytablerow .....	214	nextpose .....	247	
jobcleanup .....	215	<b>O</b>		
jobsettings .....	215	optimizefog .....	248	
<b>L</b>				
labelatom .....	217	<b>P</b>		
labelclear .....	222	partialcharge .....	248	
labelformat .....	222	pause .....	249	
labelupdate .....	222	pausecommands .....	249	
liaisonanalysis .....	223	phaseaddcustomfeature .....	249	
liaisonparameters .....	225	phaseaddligands .....	250	
liaisonselectlig .....	228	phaseaddligandsfromrun .....	251	
liaisonsettings .....	229	phasebuildqsr .....	251	
liaisonsystem .....	229	phasechooseactiveset .....	253	
ligandbond .....	231	phasecleanupstructures .....	254	
lightambient .....	231	phaseclearcentroidatoms .....	254	
lightdiffuse .....	231	phaseconfigen .....	254	
lightposition .....	232	phasecreateexcludedvolume .....	256	
lightspecular .....	232	phasedbaddconfset .....	257	
ligprep .....	233	phasedbaddligands .....	257	
ligprepstart .....	235	phasedbaddligandsfromdb .....	258	
localatoms .....	235	phasedbconfigen .....	258	
localbitset .....	236	phasedbcreatesites .....	261	
localcenter .....	236	phasedbcreatesubsetfromhitfile .....	261	
logp .....	236	phasedbcreatesubsetfromselected .....	261	
<b>M</b>				
macrodefine .....	237	phasedbcreatesubsetfromtextfile .....	262	
macrorun .....	237	phasedbdeleteligands .....	262	
makedirectory .....	238	phasedbdeletesubset .....	263	
materialmolecular .....	238	phasedbdisplayrange .....	263	
		phasedbexportligands .....	263	
		phasedbfilter .....	264	
		phasedbfilteradd .....	264	

phasedbgenerateconformers .....	265	phaselocatesites .....	284
phasedbnewfromfile .....	265	phasemarkerdump .....	284
phasedbopen .....	265	phasemarkersettings .....	285
phasedbremoveconformers .....	266	phasemarkfeature .....	295
phasedbselectextendtablerow .....	266	phaseoptions .....	295
phasedbselectonlytablerow .....	266	phasepatterndelete .....	295
phasedbselectsusubsetrow .....	267	phasepatternedit .....	296
phasedbselecttablerow .....	267	phasepatternmovedown .....	297
phasedbunselecttablerow .....	267	phasepatternmoveup .....	297
phasedeletecustomfeature .....	268	phasepatternnew .....	298
phasedeleteexcludedvolumes .....	268	phasepatternsetoptions .....	299
phasedeletehypothesis .....	268	phasepreview .....	300
phasedeletehypothesisfrombuilder .....	269	phaseqsarssearch .....	300
.....	269	phaseqsarsselecthypothesis .....	300
phasedeleteselectedligands .....	269	phaserandomtraining .....	301
phasedisplayproperty .....	269	phaserandomtrainingoptions .....	301
phaseexcludetablerow .....	269	phaserefreshfrequencytable .....	301
phaseexportalignmentsOfFile .....	270	phaseremoveligands .....	301
phaseexportconformerstoFile .....	270	phaserenameligand .....	302
phaseexportfeature .....	271	phaserescorehypotheses .....	302
phaseexporthypothesis .....	271	phaserescoreweighting .....	303
phaseexporthypothesisfrombuilder .....	271	phaseresetfeatures .....	304
.....	271	phaseruncate .....	304
phaseexportselectedalignments .....	272	phaserundele .....	305
phaseexportselectedligands .....	272	phaserunopen .....	305
phasefindmatches .....	272	phaserunrename .....	305
phasefindpharmacophores .....	274	phaserunsaveas .....	305
phasegenerateconformers .....	275	phaserunsetseed .....	306
phasehookimport .....	275	phaserunsetstereo .....	306
phasehypothesiscreateexcludedvolume .....	276	phasescorehypotheses .....	306
.....	276	phasescoreinactives .....	311
phasehypothesisdeleteexcludedvolumes .....	276	phasescoreselecthypothesis .....	311
.....	276	phasesearchformatches .....	312
phasehypothesiselectevrow .....	276	phaseselectalltablerows .....	312
phasehypothesiselectextendevrow .....	277	phaseselectevrow .....	313
.....	277	phaseselectextendevrow .....	313
phasehypothesiselectonlyevrow .....	277	phaseselectextendhypothesisrow .....	313
phasehypothesiselectrow .....	277	phaseselectextendtablerow .....	314
phasehypothesissetexcludedvolumes .....	278	phaseselecthypothesisrow .....	314
.....	278	phaseselectonlyevrow .....	314
phasehypothesissetitemask .....	278	phaseselectonlyhypothesisrow .....	315
phasehypothesissettolerances .....	278	phaseselectonlytablerow .....	315
phasehypothesistoggleinclude .....	281	phaseselecttablerow .....	315
phasehypothesisunselectevrow .....	281	phasesetactiverows .....	316
phaseimportfeature .....	281	phasesetactivity .....	316
phaseimporthypothesis .....	282	phasesetactivityproperty .....	317
phaseincludeextendtablerow .....	282	phasesetactivitythresholds .....	317
phaseincludeonlytablerow .....	282	phasesetalignmentoptions .....	318
phaseincludetablerow .....	283	phasesetexcludedvolumes .....	318
phaseinverttableselection .....	283	phasesetfrequency .....	319
phaseligprep .....	283	phasesettainingrows .....	319

phasesiteoptions .....	320	plotxyshowsidebar .....	341
phasesorttable .....	321	plotxyshowtoolbar .....	342
phasestepforward .....	321	plotxytoggledisplay .....	342
phasestepgoto .....	321	plotxyundisplay .....	342
phasetoggleactivetablerow .....	322	plotxyunlabel .....	343
phasetogglecentroidatom .....	322	plotxyunselect .....	343
phasetoggletrainingrow .....	322	plotxyupdate .....	343
phaseundisplayproperty .....	323	plotxyzoom .....	344
phaseummarkfeature .....	323	plotxyzoompan .....	344
phaseunselectevrow .....	323	posereceptor .....	345
phaseunselecthypothesisrow .....	324	potential .....	345
phaseunselecttablerow .....	324	pprep .....	347
picksize .....	324	pprepreceptorligand .....	348
place .....	325	pprepwrite .....	348
plotxyarrangecolumn .....	325	prefer .....	348
plotxyarrangerow .....	325	previouspose .....	356
plotxyarrangetiled .....	325	print .....	356
plotxyaxis .....	326	printoptions .....	356
plotxyaxisautorange .....	327	projectclose .....	358
plotxyaxisdelete .....	327	projectcopy .....	358
plotxyaxisdisplay .....	328	projectdelete .....	359
plotxyaxisrename .....	329	projectmerge .....	359
plotxycaption .....	329	projectnew .....	360
plotxycaptionposition .....	330	projectopen .....	361
plotxycopy .....	330	projectrename .....	361
plotxydelete .....	330	projectrevertopen .....	361
plotxydisplay .....	330	projectrevertsnapshot .....	361
plotxydisplaycaption .....	331	projectsaveas .....	362
plotxydisplayincluded .....	331	projectstoresnapshot .....	362
plotxydisplaylegend .....	331	projectsynchronize .....	362
plotxydisplayname .....	332	projecttablefind .....	363
plotxydisplayonly .....	332	projectupdatecoordinates .....	363
plotxydisplaypointlabels .....	332	propertycalculate .....	364
plotxydisplayselected .....	333	propertycalculatemolformula .....	365
plotxyhidemodebar .....	333	propertycalculatepickpka .....	365
plotxyhidetoolbar .....	333	propertycalculatesubstructs .....	366
plotxylabel .....	333	propertyclearvalue .....	367
plotxymove .....	334	propertycreate .....	368
plotxynew .....	335	propertydelete .....	369
plotxypan .....	335	propertygeneratecontacts .....	369
plotxypanplot .....	335	propertygeneratehbond .....	370
plotxyrename .....	336	propertymeasurementsetting .....	370
plotxyresetview .....	336	propertymove .....	370
plotxysaveimage .....	336	propertyprecision .....	371
plotxyselect .....	337	propertyrename .....	371
plotxyseries .....	337	propertyshowall .....	372
plotxyseriesdelete .....	339	propertysuperimposesetting .....	372
plotxyseriesdisplay .....	340	protassign .....	373
plotxyseriesrename .....	340	protassignresidues .....	373
plotxyseriesselect .....	340	protassignstart .....	374
plotxyseriesselectadd .....	341	protassignwrite .....	374

pspalignaddanchor .....	374	pspsequenceselect .....	394
pspaligndeleteanchor .....	374	pspsequenceviewerexport .....	394
pspaligninsertgaps .....	375	pspsetelixloopresidues .....	395
pspalignlockgaps .....	375	pspsidechainresidues .....	395
pspalignmoveleft .....	376	pspsortbbtable .....	395
pspalignmoveleftblock .....	376	pspsortfindhomologstable .....	396
pspalignmoveright .....	377	pspsortfoldtable .....	396
pspalignstructures .....	377	pspsortrbtable .....	397
pspalignunlockgaps .....	378	pspsortrbtable1 .....	397
pspbstogglehetatom .....	378	pspsortrstable .....	397
pspbuidbackbone .....	378	pspspdelete .....	398
pspbuidstructure .....	379	pspspxport .....	398
pspxcludetable1row .....	379	pspspsevert .....	398
pspxcludetablerow .....	379	pspspset .....	399
pspxportalignment .....	380	pspstepforward .....	399
pspfindfamily .....	380	pspsteppgoto .....	400
pspfindhomologs .....	380	pspstructureaddentry .....	400
pspfoldrecognitionoptions .....	380	psptemplatesetregion .....	400
pspfoldrecognitionsearch .....	381	psptmplsecstructprediction .....	401
pspimportalignment .....	381	pspunselecttable1row .....	401
pspimorthomolog .....	382	pspunselecttablerow .....	402
pspimportssp .....	382	pspupdatescores .....	402
pspincludetable1row .....	382	pyhoneval .....	403
pspincludetablerow .....	383	pythonimport .....	403
pspminimizationresidues .....	383	pythonrun .....	404
pspoptimizealignment .....	383		
psprefinebackbone .....	384		
psprefinestructure .....	384		
psprsdefaultelixloops .....	384	qikprop .....	404
psprshelixlooptoggerefine .....	384	qsarmarkerdump .....	405
psprshelixoptions .....	385	qsarmarkersettings .....	405
psprsloopoptions .....	386	qsiteion .....	407
psprunalign .....	387	qsiteresidue .....	408
pspruncreate .....	388	qsiteset .....	408
psprundelete .....	388	quit .....	410
psprunopen .....	388		
psprunrename .....	388		
psprunsaveas .....	389		
pspsecstructprediction .....	389	R	
pspselectextendtable1row .....	389	readposefile .....	411
pspselectextendtablerow .....	390	readpotential .....	411
pspselecthelixlooprow .....	390	reagentprep .....	411
pspselectonlytable1row .....	391	refinestart .....	413
pspselectonlytablerow .....	391	refinewrite .....	414
pspselectrscontext .....	392	rename .....	414
pspselectrsrefinement .....	392	repall .....	414
pspselecttable1row .....	392	repatom .....	417
pspselecttablerow .....	393	repatombonds .....	418
pspsequenceaddfile .....	393	repbond .....	418
pspsequenceaddworkspace .....	393	repdefault .....	419
pspsequencecrop .....	394	replacefromhold .....	419
		repquick .....	419

resetcsearch .....	420	surfaceextended .....	452		
residuename .....	420	surfaceextendedradiuscontext .....	453		
residuenumber .....	421	surfaceextendedradiusset .....	453		
residuerenumber .....	421	surfacemolecular .....	453		
restorepanels .....	422	surfacemolecularcontext .....	454		
retyp.....	422	surfacemolecularset .....	455		
ribbon .....	422	surfacerename .....	455		
ribbondump .....	431	surfacescheme .....	456		
ringclosure .....	431	surfacesetcomment .....	457		
rotate .....	432	surfacesetisvalue .....	457		
<b>S</b>					
saveimage .....	433	surfacesetviewas1 .....	458		
saveimageheight .....	434	surfacestyle .....	459		
saveimagewidth .....	434	surfacetransparency .....	459		
savelayout .....	434	surfacevdw .....	460		
scriptlogfile .....	435	surfacevdwcontext .....	461		
scriptrun .....	435	surfacevdwset .....	461		
searchdbconfig .....	436	surfaceviewas1set .....	462		
set .....	438	symmetrizeworkspace .....	462		
setread .....	438	system .....	463		
setwrite .....	439	<b>T</b>			
showhwstereosetup .....	439	tablealigncolumn .....	463		
showpanel .....	439	tableresizecolumn .....	464		
showpanels .....	440	tablesort .....	464		
specifiedname .....	440	tablesortall .....	465		
spotcenter .....	440	tablesortfields .....	465		
strikebuildqsar .....	441	tableunselectnonsubset .....	465		
strikeDeleteModel .....	442	tile .....	465		
strikeExportModel .....	442	toolbarmain .....	466		
strikeExtendSelectDescriptor .....	443	torsioncheck .....	466		
strikeImportModel .....	443	torsiongroup .....	467		
strikePredict .....	443	transform .....	467		
strikeSelectDescriptor .....	444	translate .....	469		
strikeSelectModel .....	444	<b>U</b>			
strikeSimilarity .....	444	undisplayatom .....	469		
strikeToggleSelectDescriptor .....	445	undisplaypose .....	469		
structalignatoms .....	445	undo .....	470		
structalignstart .....	445	ungroupentries .....	470		
substructure .....	446	unhookimport .....	470		
substructurefileread .....	447	uniqueName .....	471		
substructurefwrite .....	447	uniquePDB .....	471		
substructureshell .....	447	updateRibbons .....	472		
superimpose .....	448				
superimposeAtom .....	449				
superimposeset .....	449				
superimposesmarts .....	450				
surfacedelete .....	450				
surfacedisplay .....	451				
surfaceduplicate .....	451				

**V**

varymolecule .....	472
varytorsion .....	473
vcsaddattachment .....	473
vcsaddcorefromproject .....	474
vcsaddmincapcore .....	474
vcsaddoriginalcore .....	475
vcscanceleddockjob .....	475
vcsclearreagentfile .....	475
vcscombiexportdockingfile .....	475
vcscombiexportdockingproject .....	476
vcscombiexportoptions .....	476
vcsconfigureddocking .....	476
vcscoreoptions .....	479
vcscreativedockedlibrary .....	480
vcsdeletetattachment .....	480
vcsdeletecore .....	481
vcsdeleteresults .....	481
vcsdisplayreceptor .....	481
vcsenumerateddockoptions .....	482
vcsexcludetablerow .....	482
vcsexportdefinition .....	482
vcsexportresults .....	483
vcsimportdefinition .....	483
vcsincludeextendtablerow .....	483
vcsincludeonlytablerow .....	484
vcsincludetablerow .....	484
vcsinverttableselection .....	484
vcsoptions .....	485
vcsrefreshstructure .....	485
vcsrenameattachment .....	486
vcsrestoreresults .....	486
vcsruncombinatorial docking .....	486
vcsruncombinatorial selection .....	487
vcsruncreate .....	487
vcsrundelete .....	488
vcsruneumerated docking .....	488
vcsrunkopen .....	488
vcsrunkrename .....	488
vcsrunksaveas .....	489
vcsrunksingle docking .....	489
vcsrunksingle selection .....	489
vcssaveresults .....	490
vcsselectalltablerows .....	490
vcsselectextendtablerow .....	490
vcsselectonlytablerow .....	490

vcsselecttablerow .....	491
vcssetattachmentfile .....	491
vcssetmolecule .....	492
vcssetreagentfile .....	492
vcssingleexportdockingfile .....	492
vcssingleexportdockingproject .....	493
vcssingleexportoptions .....	493
vcssorttable .....	494
vcsstepforward .....	494
vcsstepgoto .....	494
vcsunselecttablerow .....	495
viewmatrix .....	495
viewposecontacts .....	495
viewposebonds .....	496
viewreset .....	496
viewrestore .....	496
viewsave .....	496
viewvolume .....	496
visimport .....	497
volumedelete .....	498
volumeisosurface .....	499
volumerename .....	499

**W**

workspaceselectionadd .....	500
workspaceselectionclear .....	500
workspaceselectioninvert .....	500
workspaceselectionreplace .....	501
workspaceselectionsubtract .....	501
writeangle .....	502
writecontact .....	503
writecoupling .....	503
writediagonal .....	504
writtenstance .....	505
writehbond .....	506
writeposefile .....	507

**X**

xcluster .....	507
xclusterstart .....	508
xclusterwrite .....	509

**Z**

zoom .....	509
------------	-----



# Table of Contents

<b>1</b>	<b>The Maestro Command Language .....</b>	<b>1</b>
<b>2</b>	<b>Conventions Used in this Manual .....</b>	<b>3</b>
<b>3</b>	<b>The Atom Specification Language.....</b>	<b>5</b>
3.1	Why an Atom Specification Language?.....	5
3.2	The ASL Hierarchy.....	5
3.3	Atom Specification .....	6
3.4	Operators .....	13
3.4.1	The Boolean <code>and</code> Operator .....	13
3.4.2	The Boolean <code>or</code> Operator .....	14
3.4.3	The Boolean <code>not</code> Operator .....	14
3.4.4	The <code>fillres</code> and <code>fillmol</code> Operators .....	14
3.4.5	The <code>within</code> and <code>beyond</code> Operators.....	14
3.4.6	The <code>withinbonds</code> Operator .....	15
3.4.7	The <code>beyondbonds</code> Operator .....	15
3.5	Operator Priority .....	16
3.6	Implicit Operators .....	16
3.7	Creating New Sets from Existing Ones.....	17
3.8	Special Specifications .....	17
3.9	Matching PDB Atom Names .....	18
3.10	Miscellaneous .....	18
3.10.1	Atoms not yet present in a structure .....	18
3.10.2	Aliasing.....	18
3.11	Useful Hints when using ASL with the Project Facility ...	19
3.12	ASL Examples .....	20
<b>4</b>	<b>The Entry Specification Language .....</b>	<b>25</b>
4.1	Why an Entry Selection Language?.....	25
4.2	Entry Properties .....	25
4.3	Logical Operators .....	26
4.4	Entry Property Comparisons .....	26
4.5	Examples .....	27

<b>5 Commands . . . . .</b>	<b>29</b>
1ddataset . . . . .	29
1dplot . . . . .	31
1drescale . . . . .	32
1dttable . . . . .	32
2ddataset . . . . .	32
2dplot . . . . .	33
2drescale . . . . .	34
addfromhold . . . . .	35
adjustangle . . . . .	35
adjustdihedral . . . . .	36
adjustdistance . . . . .	37
alias . . . . .	38
angle . . . . .	38
appendribbons . . . . .	38
application . . . . .	39
atom . . . . .	39
atomname . . . . .	40
attach . . . . .	40
attachmentmarkerdump . . . . .	41
attachmentmarkersettings . . . . .	41
autosetup . . . . .	44
beginundoblock . . . . .	45
bmincomfile . . . . .	45
bond . . . . .	45
bondorder . . . . .	46
bondtonew . . . . .	46
buildoptions . . . . .	47
calcenergy . . . . .	47
canonicalname . . . . .	48
cascadepanels . . . . .	48
cd . . . . .	48
cellsmarkerdump . . . . .	49
cellsmarkersettings . . . . .	49
centeratom . . . . .	52
centerbond . . . . .	53
centercoordinates . . . . .	53
centroid . . . . .	54
centroidatom . . . . .	54
cglidedockconstraintposition . . . . .	54
cglidedockconstraintregion . . . . .	55
chainname . . . . .	57
changedirectory . . . . .	57
chdir . . . . .	57
chiralatom . . . . .	58

clip .....	58
clusteratom .....	58
clusterheavyatoms .....	59
clusterset .....	59
clustertorsion .....	60
coloratom .....	60
colorscheme .....	61
combilibenum .....	61
combilibenumaddattachment .....	61
combilibenumclearreagentfile .....	62
combilibenumdeleteattachment .....	62
combilibenumexportdefinition .....	62
combilibenumimportdefinition .....	63
combilibenumoptions .....	63
combilibenumrefreshstructure .....	64
combilibenumrenameattachment .....	64
combilibenumselectextendtablerow .....	64
combilibenumselectonlytablerow .....	65
combilibenumselecttablerow .....	65
combilibenumsetmolecule .....	66
combilibenumsetreagentfile .....	66
combilibenumunselecttablerow .....	66
compareatom .....	67
compareset .....	67
confelim .....	68
confelimstart .....	69
confelimwrite .....	69
confgenltsearch .....	69
confgenmini .....	71
confgenpotential .....	73
confgenreadpotential .....	75
confgenstart .....	75
confgenwrite .....	75
confsearch .....	75
connect .....	78
connectfuseatom .....	78
constrainedangle .....	78
constrainedatom .....	79
constraineddist .....	80
constrainedset .....	81
constrainedtorsion .....	81
contactcriteria .....	82
contactset1 .....	83
contactset2 .....	84
coupling .....	84

createlibrary	85
createpropsubset	85
createsubset	85
defaultfc	85
delete	86
deleteproperty	87
dialogoptions	87
dihedral	87
dihedraldrive	88
displayatom	88
displayonlyatom	89
displayonlypose	89
displayopt	89
displaypose	93
distance	93
distancecheck	93
dynamics	94
ecalc	95
ejob	95
elementlabels	97
embrace	97
embracecsearch	98
endundoblock	100
energykill	100
energymonitor	101
energysleep	101
energystart	102
energystop	102
energytask	102
energyupdate	103
energyupdatejobstatus	103
energywake	104
enhance3d	104
entrycombine	104
entrycopyprop	105
entrydelete	106
entrydisassociate	106
entrydragselection	107
entryduplicate	107
entryexport	108
entryexportspreadsheet	110
entryextendselection	111
entryextendwsinclude	111
entrygroupcollapse	112
entrygroupcreate	112

entrygroupdelete .....	112
entrygroupduplicate .....	113
entrygroupexpand .....	113
entrygroupexpandonly .....	113
entrygroupextendselection .....	114
entrygroupmove .....	114
entrygrouprename .....	115
entrygroupselect .....	115
entrygroupselectonly .....	115
entrygroupsettitle .....	116
entrygroupungroup .....	116
entrygroupunselect .....	116
entrygroupwsexclude .....	117
entrygroupwsinclude .....	117
entrygroupwsincludeonly .....	118
entryimport .....	118
entryimportspreadsheet .....	120
entryimportvibration .....	121
entryinvertselection .....	121
entrymovetogroup .....	122
entryrename .....	122
entryselect .....	123
entryselectall .....	123
entryselectonly .....	123
entryselectonlyrow .....	124
entryselectrandom .....	124
entryselectrow .....	125
entrysetprop .....	125
entrysettitle .....	126
entryshowall .....	126
entrytable .....	126
entryunselect .....	127
entryunselectall .....	127
entryunselectrow .....	127
entrywscreate .....	127
entrywsexclude .....	128
entrywsexcludenotfixed .....	128
entrywsinclude .....	129
entrywsincludelock .....	129
entrywsincludeonly .....	129
epik .....	130
epikstart .....	132
eplayergoto .....	132
eplayergotofirst .....	132
eplayergotolast .....	133

eplayersettings	133
eplayerstepahead	135
eplayerstepback	135
errorcheck	135
excludedvolumesmarkersettings	136
extenddisplaytopose	137
fileread	138
filewrite	140
filter	142
find	142
fit	145
forcefield	145
forcefieldbend	145
forcefielddele	146
forcefieldgbsolv	147
forcefieldimproper	147
forcefieldsasolv	148
forcefieldstretch	149
forcefieldtorsion	149
forcefieldvdw	150
forcefieldview	151
forcefieldwilson	151
formalcharge	152
frament	152
fragmenttype	153
frozenatom	153
frozensest	154
fuse	154
geometrycleanup	154
glideactivesiteres	155
glideactivesiteset	155
glidecalbboxfromligand	155
glideconstraintatomlabel	156
glideconstraintposition	156
glideconstraintregion	157
glideconstraintregionactivecell	157
glideconstraintregiongrow	157
glideconstraintregioninvisible	158
glideconstraintregionshrink	158
glideconstraintregionvisible	158
glidedisplayreceptor	159
glidedockconstraintposition	159
glidedockconstraintregion	160
glidedockregionnumatoms	161
glidedockregionselect	162

glidedockregionunselect	162
glidegridhydrophobic	163
glideligand	163
glideligandfile	166
glideligandgrid	166
glidelocatehydrophobiccells	168
glideoutput	168
glidereceptorligand	169
gliderefenceligand	169
glidescoring	170
glidesettings	172
glidesite	175
grow	177
growbond	177
growbond2	178
growdirection	178
growname	178
happily	179
hbondcriteria	179
hbondset1	180
hbondset2	180
help	180
helpauto	181
helpcategory	181
helpsearch	181
helptopic	182
hideentries	183
hidepanel	183
hidepanels	183
hideproperty	183
historyvisible	184
hold	191
hppmap	191
hppmapbox	192
hppmapset	193
hppmapstart	193
hppmapwrite	193
htreat	193
hydrogenapply	194
hydrogentreat	194
impactbufferedatom	194
impactbufferedset	195
impactconstraints	195
impactcontinuumsolvent	196
impactdynamics	197

impactfastmultipole	199
impactfrozenatom	200
impactfrozenset	200
impacthybridmc	200
impactjob	201
impactmdparams	202
impactminimization	204
impactperiodicboundary	206
impactpotential	207
impactstart	209
impacttask	209
impacttruncation	209
impactwrite	210
invert	210
invertset	210
jaguarassignatomnames	211
jaguarimportgeometry	211
jaguarinputfilesjob	212
jaguarjob	213
jaguarselectextendtablerow	213
jaguarselectonlytablerow	214
jobcleanup	214
jobsettings	215
kill	217
labelatom	217
labelclear	222
labelformat	222
labelupdate	222
liaisonanalysis	223
liaisonparameters	225
liaisonselectlig	228
liaisonsettings	228
liaisonsystem	229
ligandbond	231
lightambient	231
lightdiffuse	231
lightposition	232
lightspecular	232
ligprep	233
ligprepstart	235
localatoms	235
localbitset	235
localcenter	236
logfile	236
logp	236

macrodefine	237
macrорun	237
makedirectory	238
materialmolecular	238
mcsd	239
mini	239
minienergy	240
minta	241
monitor	242
monitorangle	242
monitordistance	242
monitorhbond	243
monitorsetsurf	244
monitorsettings	244
monitorsurf	244
monitortorsion	245
mouse	245
move	246
multiplemini	246
mutate	247
nextpose	247
optimizefog	248
partialcharge	248
pause	248
pausecommands	249
phaseaddcustomfeature	249
phaseaddligands	250
phaseaddligandsfromrun	250
phasebuildqsar	251
phasechooseactiveset	253
phasecleanupstructures	254
phaseclearcentroidatoms	254
phaseconfgen	254
phasecreateexcludedvolume	256
phasedbaddconfset	257
phasedbaddligands	257
phasedbaddligandsfromdb	258
phasedbconfigen	258
phasedbcreatesites	261
phasedbcreatesubsetfromhitfile	261
phasedbcreatesubsetfromselected	261
phasedbcreatesubsetfromtextfile	262
phasedbdeleteligands	262
phasedbdeletesubset	262
phasedbdisplayrange	263

phasedbexportligands	263
phasedbfilter	264
phasedbfilteradd	264
phasedbgenerateconformers	264
phasedbnewfromfile	265
phasedbopen	265
phasedbremoveconformers	266
phasedbselectextendtablerow	266
phasedbselectonlytablerow	266
phasedbselectsubsetrow	267
phasedbselecttablerow	267
phasedbunselecttablerow	267
phasedeletecustomfeature	268
phasedeleteexcludedvolumes	268
phasedeletehypothesis	268
phasedeletehypothesisfrombuilder	268
phasedeleteselectedligands	269
phasedisplayproperty	269
phaseexcludetablerow	269
phaseexportalignmentsOfFile	270
phaseexportconformerstoFile	270
phaseexportfeature	270
phaseexportHypothesis	271
phaseexportHypothesisFromBuilder	271
phaseexportSelectedAlignments	272
phaseexportSelectedLigands	272
phasefindmatches	272
phasefindpharmacophores	274
phasegenerateconformers	275
phasehookimport	275
phasehypothesiscreateexcludedvolume	276
phasehypothesisdeleteexcludedvolumes	276
phasehypothesiselectevrow	276
phasehypothesiselectextendrow	276
phasehypothesiselectonlyevrow	277
phasehypothesiselectrow	277
phasehypothesissetexcludedvolumes	277
phasehypothesissetitemask	278
phasehypothesissettolerances	278
phasehypothesistoggleinclude	280
phasehypothesisunselectevrow	281
phaseimportfeature	281
phaseimportHypothesis	282
phaseincludeextendtablerow	282
phaseincludeonlytablerow	282

phaseincludetablerow	283
phaseinverttableselection	283
phaseligprep	283
phaselocatesites	284
phasemarkerdump	284
phasemarkersettings	285
phasemarkfeature	294
phaseoptions	295
phasepatterndelete	295
phasepatternedit	296
phasepatternmovedown	297
phasepatternmoveup	297
phasepatternnew	298
phasepatternsetoptions	299
phasepreview	300
phaseqssarsearch	300
phaseqssarselecthypothesis	300
phaserandomtraining	300
phaserandomtrainingoptions	301
phaserefreshfrequencytable	301
phaseremoveligands	301
phaserenamelingand	302
phaserescorehypotheses	302
phaserescoreweighting	302
phaseresetfeatures	304
phaseruncreate	304
phaserundelete	304
phaserunopen	305
phaserunrename	305
phaserunsaveas	305
phaserunsetseed	306
phaserunsetstereo	306
phasescorehypotheses	306
phasescoreinactives	311
phasescoreselecthypothesis	311
phasesearchformatches	312
phaseselectalltablerows	312
phaseselectevrow	312
phaseselectextendrow	313
phaseselectextendhypothesisrow	313
phaseselectextendtablerow	313
phaseselecthypothesisrow	314
phaseselectonlyevrow	314
phaseselectonlyhypothesisrow	314
phaseselectonlytablerow	315

phaseselectablerow	315
phasesetactiverows	315
phasesetactivity	316
phasesetactivityproperty	316
phasesetactivitythresholds	317
phasesetalignmentoptions	317
phasesetexcludedvolumes	318
phasesetfrequency	319
phasesetraininggrows	319
phasesiteoptions	320
phasesorttable	320
phasestepforward	321
phasestepgoto	321
phasetoggleactivetablerow	322
phasetogglecentroidatom	322
phasetoggletrainingrow	322
phaseundisplayproperty	323
phaseunmarkfeature	323
phaseunselectevrow	323
phaseunselecthypothesisrow	324
phaseunselectablerow	324
picksize	324
place	325
plotxyarrangecolumn	325
plotxyarrangerow	325
plotxyarrangetiled	325
plotxyaxis	326
plotxyaxisautorange	327
plotxyaxisdelete	327
plotxyaxisdisplay	328
plotxyaxisrename	328
plotxycaption	329
plotxycaptionposition	329
plotxycopy	330
plotxydelete	330
plotxydisplay	330
plotxydisplaycaption	330
plotxydisplayincluded	331
plotxydisplaylegend	331
plotxydisplayname	332
plotxydisplayonly	332
plotxydisplaypointlabels	332
plotxydisplayselected	332
plotxyhidemodebar	333
plotxyhidetoolbar	333

plotxylabel	333
plotxymove	334
plotxynew	335
plotxypan	335
plotxypanplot	335
plotxyrename	336
plotxyresetview	336
plotxysaveimage	336
plotxyselect	337
plotxyseries	337
plotxyseriesdelete	339
plotxyseriesdisplay	339
plotxyseriesrename	340
plotxyseriesselect	340
plotxyseriesselectadd	341
plotxyshowsidebar	341
plotxyshowtoolbar	342
plotxytoggledisplay	342
plotxyundisplay	342
plotxyunlabel	342
plotxyunselect	343
plotxyupdate	343
plotxyzoom	344
plotxyzoompan	344
posereceptor	344
potential	345
pprep	347
pprepreceptorligand	347
pprepwrite	348
prefer	348
previouspose	356
print	356
printoptions	356
projectclose	358
projectcopy	358
projectdelete	359
projectmerge	359
projectnew	360
projectopen	360
projectrename	361
projectrevertopen	361
projectrevertsnapshot	361
projectsaveas	362
projectstoresnapshot	362
projectsynchronize	362

projecttablefind	362
projectupdatecoordinates	363
propertycalculate	364
propertycalculatemolformula	365
propertycalculatepickpka	365
propertycalculatesubstructs	366
propertyclearvalue	367
propertycreate	368
propertydelete	369
propertygeneratecontacts	369
propertygeneratehbond	370
propertymeasurementsetting	370
propertymove	370
propertyprecision	371
propertyrename	371
propertyshowall	372
propertysuperimposesetting	372
protassign	373
protassignresidues	373
protassignstart	373
protassignwrite	374
pspalignaddanchor	374
pspaligndeleteanchor	374
pspaligninsertgaps	375
pspalignlockgaps	375
pspalignmoveleft	375
pspalignmoveleftblock	376
pspalignmoveright	377
pspalignstructures	377
pspalignunlockgaps	378
pspbstogglehetatom	378
pspbuildbackbone	378
pspbuiltstructure	378
pspexcludedtable1row	379
pspexcludetablerow	379
pspexportalignment	380
pspfindfamily	380
pspfindhomologs	380
pspfoldrecognitionoptions	380
pspfoldrecognitionsearch	381
pspimportalignment	381
pspimportorthomolog	381
pspimportssp	382
pspincludetable1row	382
pspincludetablerow	383

pspminimizationresidues . . . . .	383
pspoptimizealignment . . . . .	383
psprefinebackbone . . . . .	384
psprefinestructure . . . . .	384
psprsdefaulthelixloops . . . . .	384
psprshelixlooptogglerefine . . . . .	384
psprshelixoptions . . . . .	385
psprsloopoptions . . . . .	386
psprunalign . . . . .	387
pspruncreate . . . . .	387
psprundelete . . . . .	388
psprunopen . . . . .	388
psprunrename . . . . .	388
psprunsaveas . . . . .	389
pspsecstructprediction . . . . .	389
pspselectextendtable1row . . . . .	389
pspselectextendtablerow . . . . .	390
pspselecthelixlooprow . . . . .	390
pspselectonlytable1row . . . . .	391
pspselectonlytablerow . . . . .	391
pspselectrscontext . . . . .	391
pspselectrsrefinement . . . . .	392
pspselecttable1row . . . . .	392
pspselecttablerow . . . . .	392
pspsequenceaddfile . . . . .	393
pspsequenceaddworkspace . . . . .	393
pspsequencecrop . . . . .	393
pspsequenceselect . . . . .	394
pspsequenceviewerexport . . . . .	394
pspsethelixloopresidues . . . . .	395
pspsidechainresidues . . . . .	395
pspsortbbtable . . . . .	395
pspsortfindhomologstable . . . . .	396
pspsortfoldtable . . . . .	396
pspsortrtable . . . . .	396
pspsortrtable1 . . . . .	397
pspsortrstable . . . . .	397
pspsspdelete . . . . .	397
pspsspexport . . . . .	398
pspssprevert . . . . .	398
pspsspset . . . . .	399
pspstepforward . . . . .	399
pspstepgoto . . . . .	400
pspstructureaddentry . . . . .	400
psptemplatesetregion . . . . .	400

psptmplsecstructprediction . . . . .	401
pspunselecttable1row . . . . .	401
pspunselecttablerow . . . . .	402
pspupdatescores . . . . .	402
pyeval . . . . .	402
pyimp . . . . .	403
pyrun . . . . .	403
pythoneval . . . . .	403
pythonimport . . . . .	403
pythonrun . . . . .	404
qikprop . . . . .	404
qsarmarkerdump . . . . .	404
qsarmarkersettings . . . . .	405
qsiteion . . . . .	407
qsiteresidue . . . . .	407
qsiteset . . . . .	408
quit . . . . .	410
read . . . . .	410
readposefile . . . . .	411
readpotential . . . . .	411
reagentprep . . . . .	411
refinestart . . . . .	413
refinewrite . . . . .	414
rename . . . . .	414
renameproperty . . . . .	414
repall . . . . .	414
repatom . . . . .	417
repatombonds . . . . .	418
repbond . . . . .	418
repdefault . . . . .	418
replacefromhold . . . . .	419
repquick . . . . .	419
resetcsearch . . . . .	420
residuename . . . . .	420
residuenumber . . . . .	421
residuerenumber . . . . .	421
restorepanels . . . . .	422
retype . . . . .	422
ribbon . . . . .	422
ribbondump . . . . .	431
ringclosure . . . . .	431
rotate . . . . .	432
run . . . . .	433
saveimage . . . . .	433
saveimageheight . . . . .	434

saveimagewidth	434
savelayout	434
scriptlogfile	435
scriptrun	435
searchdbconfig	436
set	438
setread	438
setwrite	439
showhwstereosetup	439
showpanel	439
showpanels	440
sleep	440
specifiedname	440
spotcenter	440
stop	441
strikebuildqsar	441
strikedeletemodel	442
strikeexportmodel	442
strikeextendselectdescriptor	443
strikeimportmodel	443
strikepredict	443
strikeselectdescriptor	444
strikeselectmodel	444
strikesimilarity	444
striketoggleselectdescriptor	445
structalignatoms	445
structalignstart	445
substructure	446
substructurefile	446
substructurefileread	446
substructurefilewrite	447
substructureshell	447
superimpose	448
superimposeatom	449
superimposeset	449
superimposesmarts	450
surfacedelete	450
surfacedisplay	451
surfaceduplicate	451
surfaceextended	452
surfaceextendedradiuscontext	453
surfaceextendedradiusset	453
surfacemolecular	453
surfacemolecularcontext	454
surfacemolecularset	455

surfacerename	455
surfacescheme	456
surfacesetcomment	457
surfacesetisvalue	457
surfacesetviewasl	458
surfacestyle	459
surfacetransparency	459
surfacevdw	460
surfacevdwcontext	461
surfacevdwset	461
surfaceviewaslset	462
symmetrizeworkspace	462
system	463
tablealigncolumn	463
tableresizecolumn	463
tablesort	464
tablesortall	464
tablesortfields	465
tableunselectnsubset	465
tile	465
toolbarmain	466
torsioncheck	466
torsiongroup	467
transform	467
translate	468
undisplayatom	469
undisplaypose	469
undo	470
ungroupentries	470
unhookimport	470
uniquepname	471
uniquepdb	471
update	471
updateribbons	471
varymolecule	472
varytorsion	473
vcsaddattachment	473
vcsaddcorefromproject	474
vcsaddmincapcore	474
vcsaddoriginalcore	474
vscanceleddockjob	475
vsclearreagentfile	475
vcscombiexportdockingfile	475
vcscombiexportdockingproject	475
vcscombiexportoptions	476

vcsconfigureddocking	476
vcscoreoptions	479
vcscreativedockedlibrary	480
vcsdeleteattachment	480
vcsdeletetcore	481
vcsdeleteresults	481
vcsdisplayreceptor	481
vcsenumerateddockoptions	481
vcsexcludetablerow	482
vcsexportdefinition	482
vcsexportresults	483
vcsimportdefinition	483
vcsincludeextendtablerow	483
vcsincludeonlytablerow	484
vcsincludetablerow	484
vcsinverttableselection	484
vcsoptions	484
vcsrefreshstructure	485
vcsrenameattachment	485
vcsrestoreresults	486
vcsruncombinatorialdocking	486
vcsruncombinatorialselection	487
vcsruncreate	487
vcsrundelete	487
vcsrunenumerateddocking	488
vcsrunopen	488
vcsrurname	488
vcsrunsaveas	489
vcsrungledocking	489
vcsrungingleselection	489
vcssaveresults	489
vcsselectalltablerows	490
vcsselectextendtablerow	490
vcsselectonlytablerow	490
vcsselecttablerow	491
vcssetattachmentfile	491
vcssetmolecule	492
vcssetreagentfile	492
vcssingleexportdockingfile	492
vcssingleexportdockingproject	493
vcssingleexportoptions	493
vcssorttable	494
vcsstepforward	494
vcsstepgoto	494
vcsunselecttablerow	495

viewmatrix	495
viewposecontacts	495
viewposebonds	495
viewreset	496
viewrestore	496
viewsave	496
viewvolume	496
visimport	497
volumedelete	498
volumeisosurface	498
volumerename	499
wake	500
workspaceselectionadd	500
workspaceselectionclear	500
workspaceselectioninvert	500
workspaceselectionreplace	501
workspaceselectionsubtract	501
write	501
writeangle	502
writecontact	502
writecoupling	503
writedihedral	504
writedistance	505
writehbond	506
writeposefile	507
xcluster	507
xclusterstart	508
xclusterwrite	508
zoom	509
<b>Command Index</b>	<b>511</b>